REMINDERS, UPDATES

- Part 2 (bonus) of lab 2 will be released tomorrow
- Midterm next week – Thursday, June 3, during lecture (2.5hrs)
  - On Zoom, cameras ON
  - One design problem, based on lectures & labs so far
  - Will need to use Quartus & Modelsim
- Quiz weight (5%) moved to final exam (becomes 30%)

CPEN 311: Digital Systems Design

FPGA Architecture and CAD

You have now learned enough Verilog that you can design ANYTHING

What can we optimize?

Circuit Area: Leads to smaller, cheaper chips, cheaper products
(or more profit margin)

Circuit Speed: Faster frequencies, more work in each clock cycle
-> Makes new applications possible!
-> Today, machine learning is limited by processing speed

Circuit Power: Lighter (smaller battery), longer run-time
-> Reduce heat → Dark Silicon

To understand how to optimize designs, we need to understand how designs are implemented. We will focus on FPGAs. (take ELEC 402 and 403 to talk about custom chips).

Learning objective for this slide set:

- Understand how designs are implemented on an FPGA
- For some of you, this is a repeat, for some it is new

Once you understand how computation occurs on an FPGA, we can talk about how to improve the designs.

Learning Objectives

After this slide set in lectures you should be able to:

Be able to talk intelligibly about the difference between hardware and software

Understand the basic steps the tools use to convert Verilog to hardware

Understand what is inside an FPGA

Understand the two sources of flexibility in an FPGA:

1. Logic block behaviour can be configured
2. Connections between logic blocks can be configured

Show how a digital circuit can be implemented using FPGA logic blocks

Semiconductor Manufacturing

- 3 ways to make a chip
  - Fully custom design ($$$$$$$$)
  - Standard cells ($$$$)
  - FPGAs ($)

- Need advanced software tools, aka
  - Electronic Design Automation tools (EDA)
  - Computer-Aided Design tools (CAD)

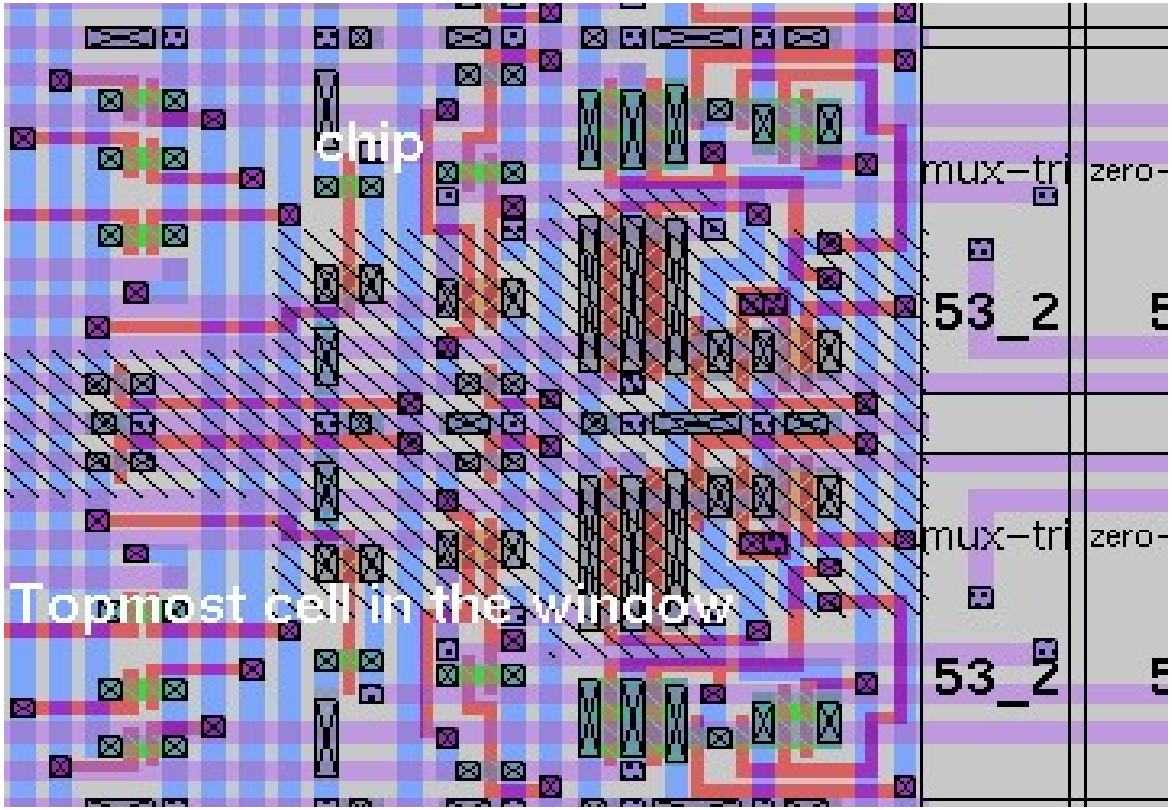
- Today, you write Verilog at the logic level
  - But you can also write it at transistor level
  - Verilog AMS for analog/mixed-signal design
  - Run analog circuit simulators like SPICE

Fully custom integrated circuits

Transistors & wires are drawn (mostly squares/lines/rectangles)

Representing different manufacturing steps

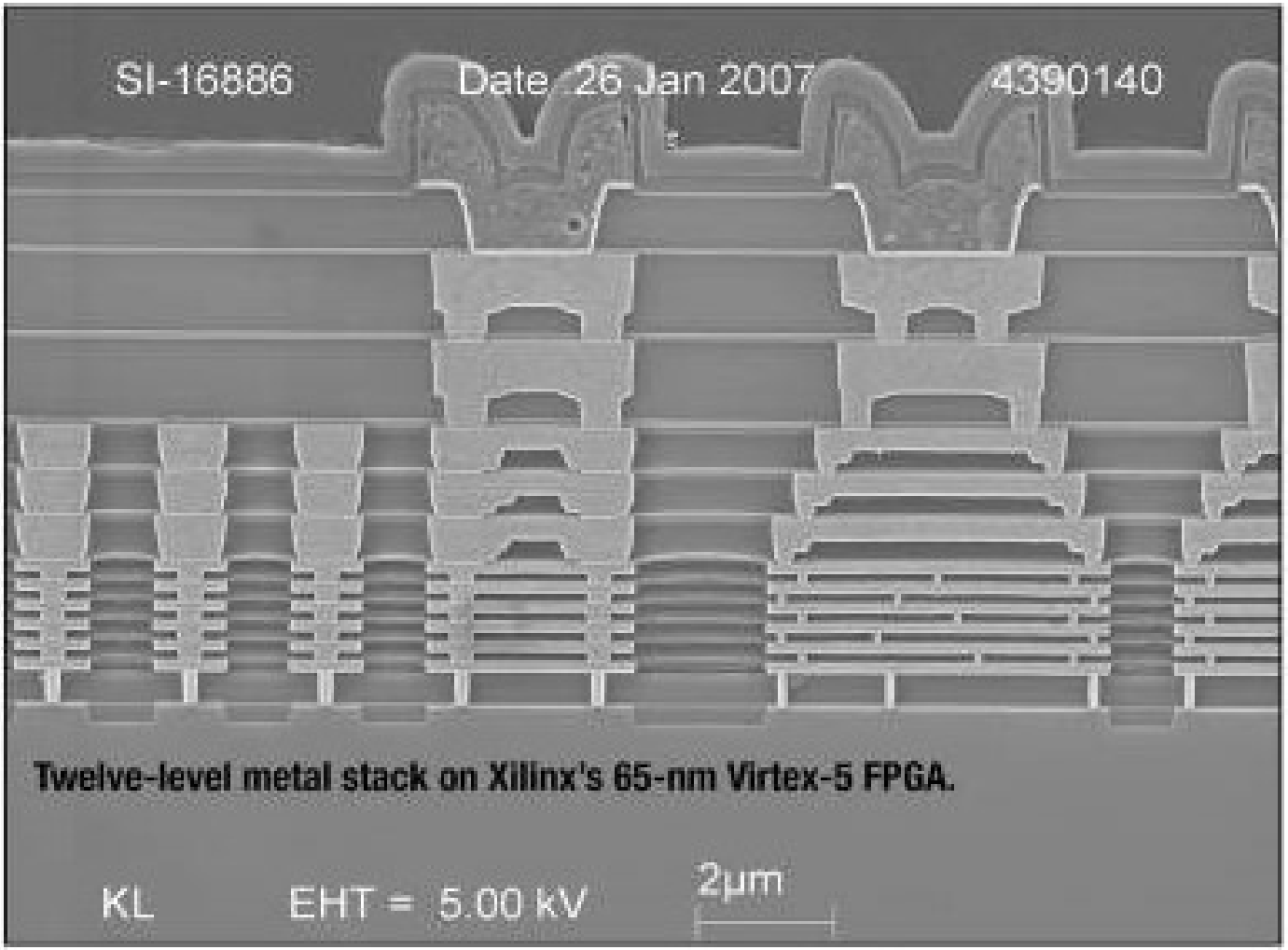
Logic gates implemented using individual transistors and wires



This is from a previous EECE 479/ELEC 402 project

Chip Cross-section Showing Manufacturing Layers

(semiconductor transistors are in lowest layer and are manufactured first, the rest is metal and insulator)



Fully custom integrated circuits

Advantages: best density and speed

Challenges: very long design time

- “Time to market” is everything
- With advanced technologies, interactions between devices on the chip are becoming increasingly hard to model

Today: only really high-end chips are designed this way

Standard Cell Design

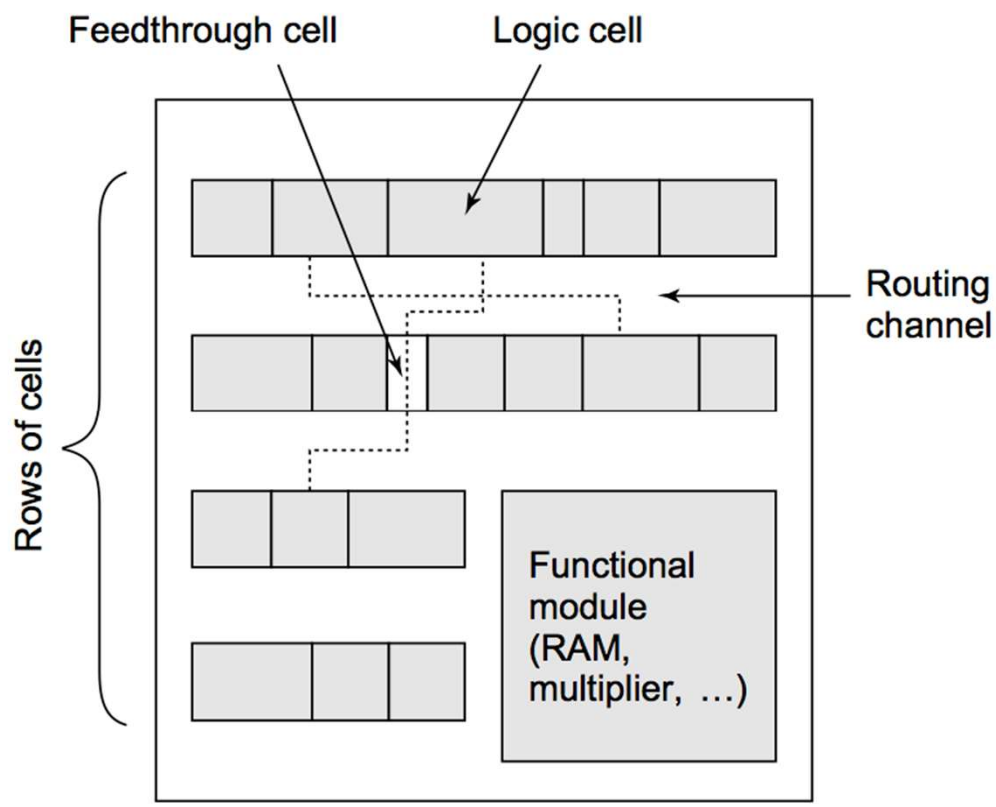
Similar to full custom

- But uses a library of cells (each implement different logic gates)
- Cell library is designed by the manufacturer (fab)
- Many variants of the same cell (small NAND, big NAND, bigger NAND, etc.)

Cells can have different width,
but **all cells have same height**
(hence Standard Cells)

Cells are placed on rows on Si substrate

- Rows are spaced apart for routing the wires (interconnect), gaps vary
- Cells within a row sometimes have gaps for wires (route-through or feedthrough), gaps vary



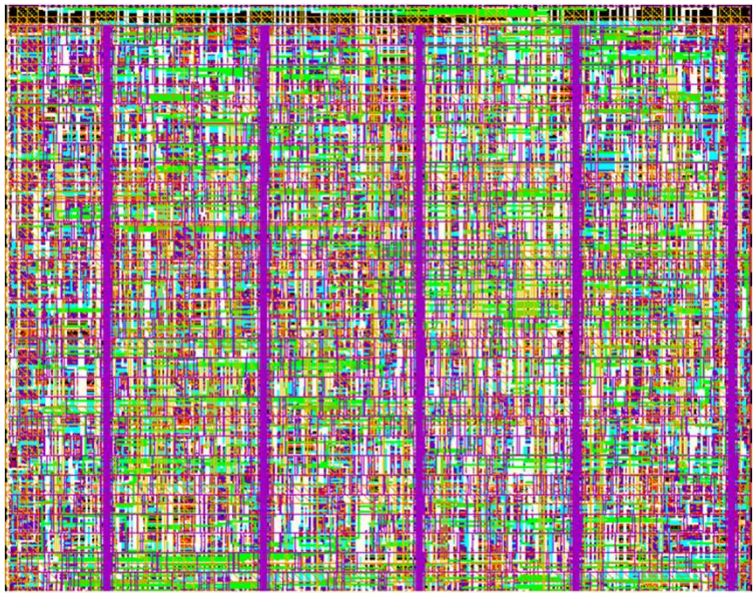
Standard Cell Design:

Used only for high-speed or low-power applications

Very expensive, and time consuming (> $2M just for the mask costs)

Very high re-spin cost

More economical than full custom



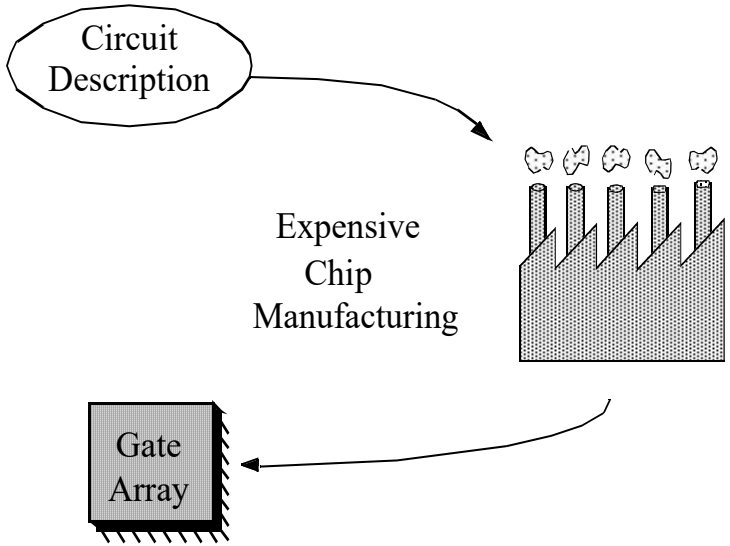
Cell-structure hidden under interconnect layers

If you want to learn more about custom chip / standard cell design, take ELEC 402 and possibly ELEC 403 next year.

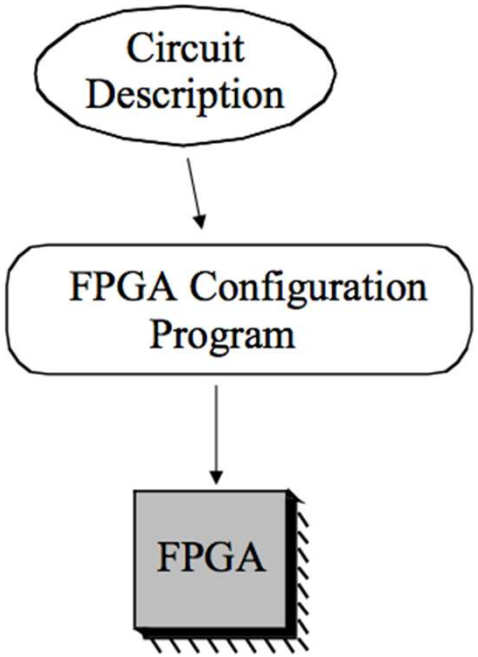
FPGAs: Field-Programmable Gate Arrays

A flexible integrated circuit that can be used to implement any digital hardware circuit, in the field, in seconds.

With a full custom chip:



With an FPGA:



Advantages of FPGAs:

1. "Instant Manufacturability": reduces time to market
2. Cheaper for small volumes because you don't need to pay for fabrication

   - means you don't need to be a big company to make a chip
3. Relaxes Designers -> relaxed designers live longer!

Disadvantages of FPGAs:

1. Slower than gate arrays or custom chips
2. Can not get as much circuitry on a single chip

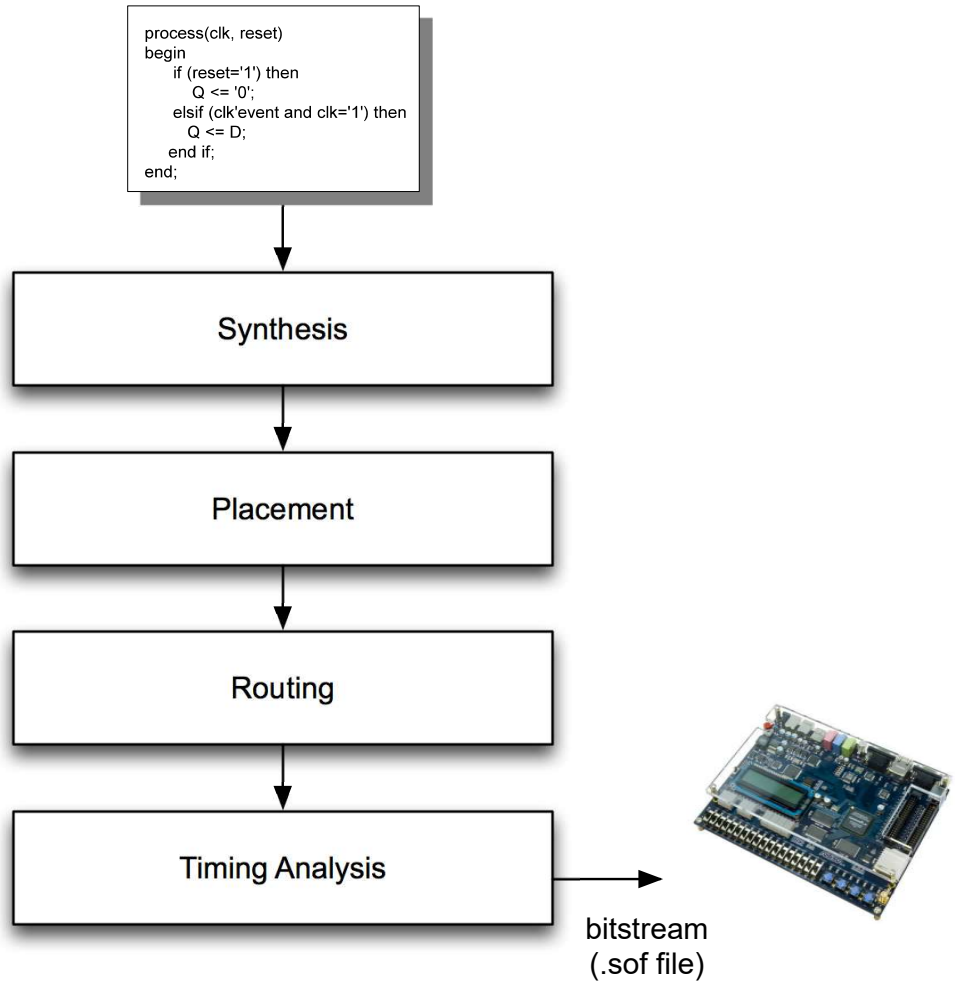
   Today: ~ 2M gates is the best you can do

   ~ 600 MHz is about as fast as you can get
3. For large volumes, it can be more expensive than gate arrays and custom chips

FPGA Compilation: Verilog -> Hardware

When you hit compile, the Quartus Prime tool converts your Verilog code to a .sof file that can be programmed on the FPGA.

This is done through a number of transforms that we will talk about in this slide set.



FPGA CAD and Architecture

The main focus of this slide set is:

- What goes on inside the compiler (CAD, computer-aided design)
- What the architecture of an FPGA looks like

Do we care? The tools shield us pretty well from the internals.

But, it helps to understand what is going on under-the-hood:

- You can better optimize your design if you understand how it is being implemented (smaller, faster, less power → more $$$)
- It can be helpful during debugging
- Important to understand how an FPGA is built when you are selecting an FPGA for a project

FPGAs are available from several companies

Intel and Xilinx (Xilinx is a bit bigger than Intel's FPGA biz)

THE WALL STREET JOURNAL.

English Edition ▾ | Print Edition | Video | Podcasts | Latest Headlines

Home World U.S. Politics Economy **Business** Tech Markets Opinion Life & Arts Real Estate WSJ. Magazine

◆ WSJ NEWS EXCLUSIVE | BUSINESS

AMD Is in Advanced Talks to Buy Xilinx

Deal that could be worth more than $30 billion would mark the latest big tie-up in the rapidly consolidating industry

Microchip (formerly Microsemi/Actel): non-volatile configuration, instant power-on, very low power, radiation hardened

Lattice Semiconductor: low-cost/consumer

Achronix: networking FPGAs (formerly manufactured by Intel)

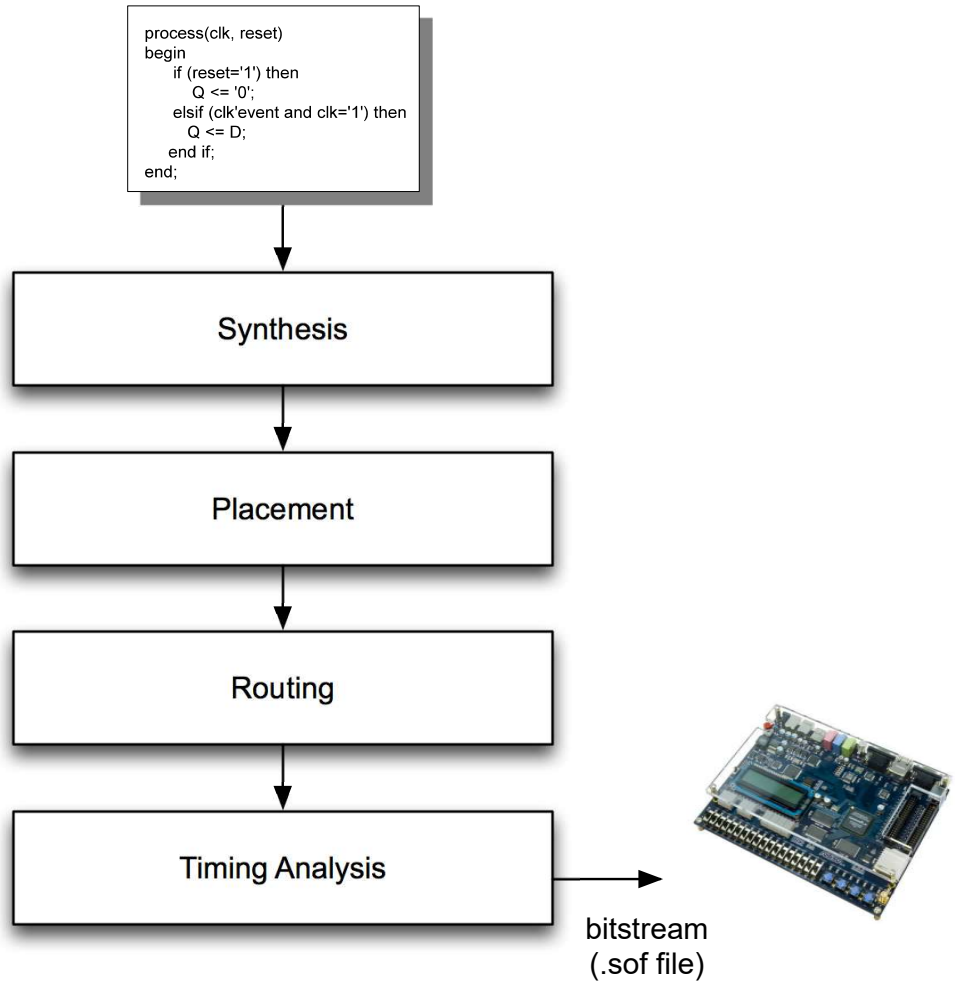
FlexLogix, Efinix: startups: embedded FPGAs

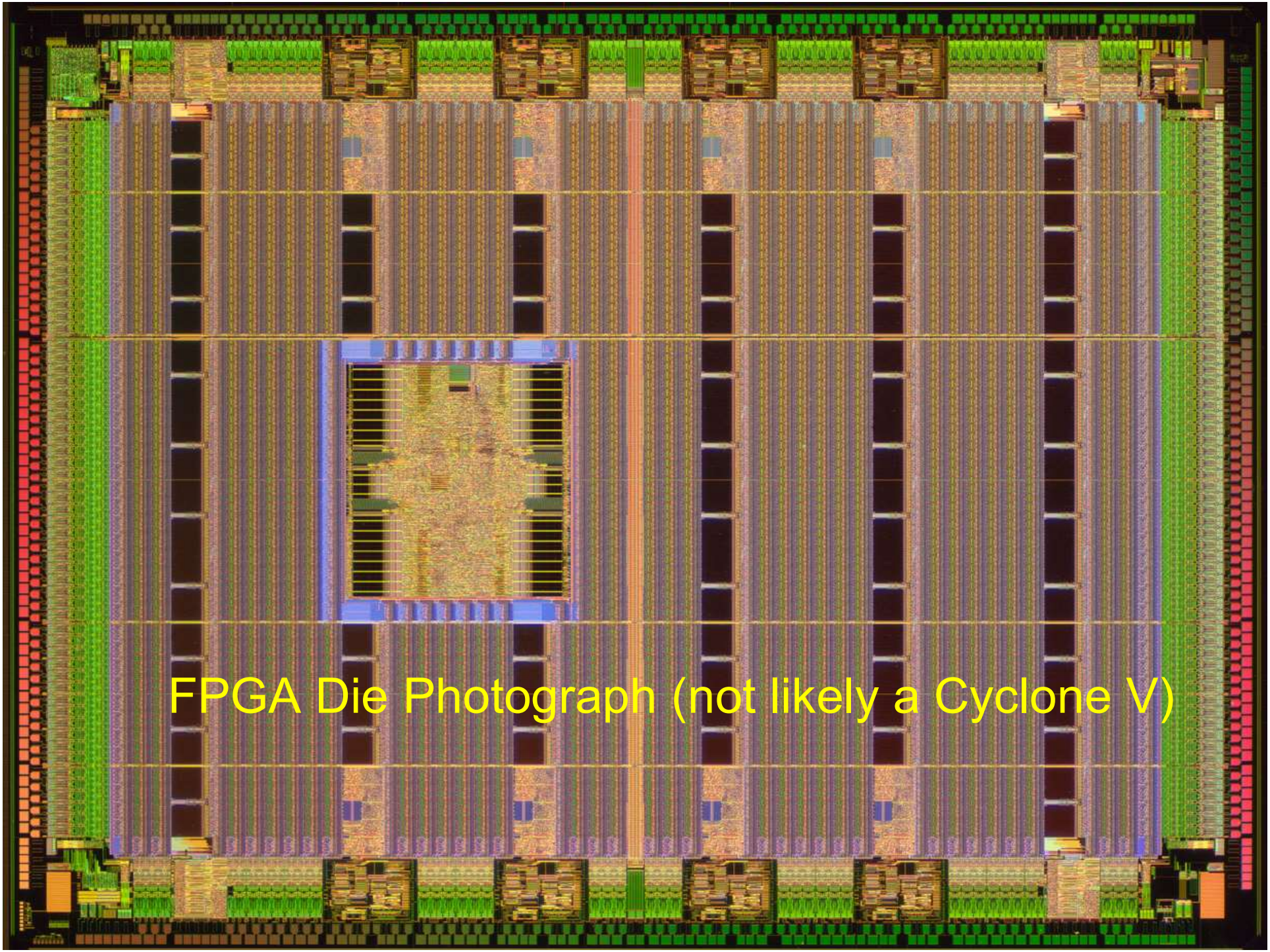
Each company has its own associated software tool suite (Intel Quartus Prime, Xilinx Vivado, Microsemi Libero, …)

FPGA Compilation:

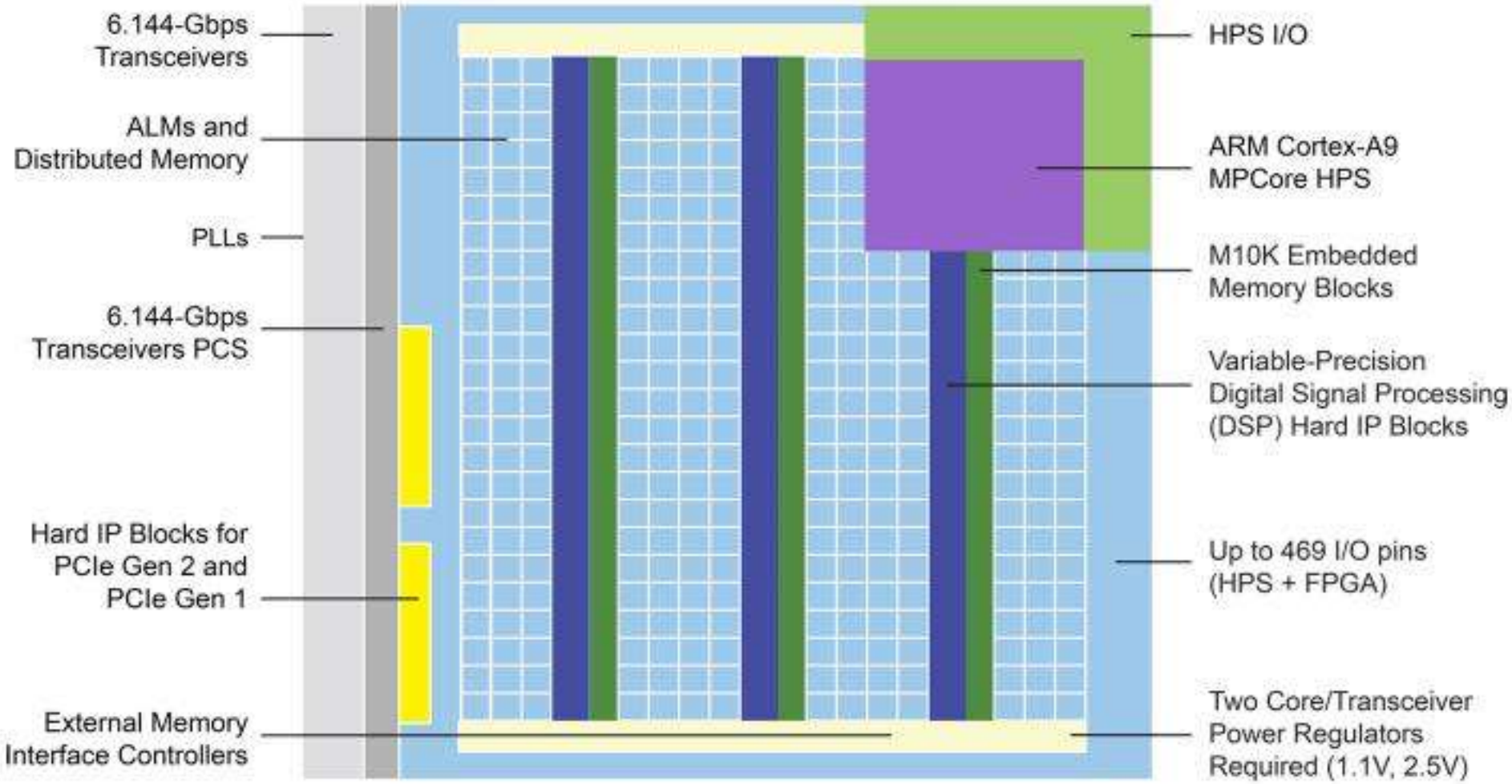
When you hit compile, the Quartus II tool converts your Verilog code to a .sof file that can be programmed on the FPGA.

This is done through a number of transforms that we will talk about in the next few slides

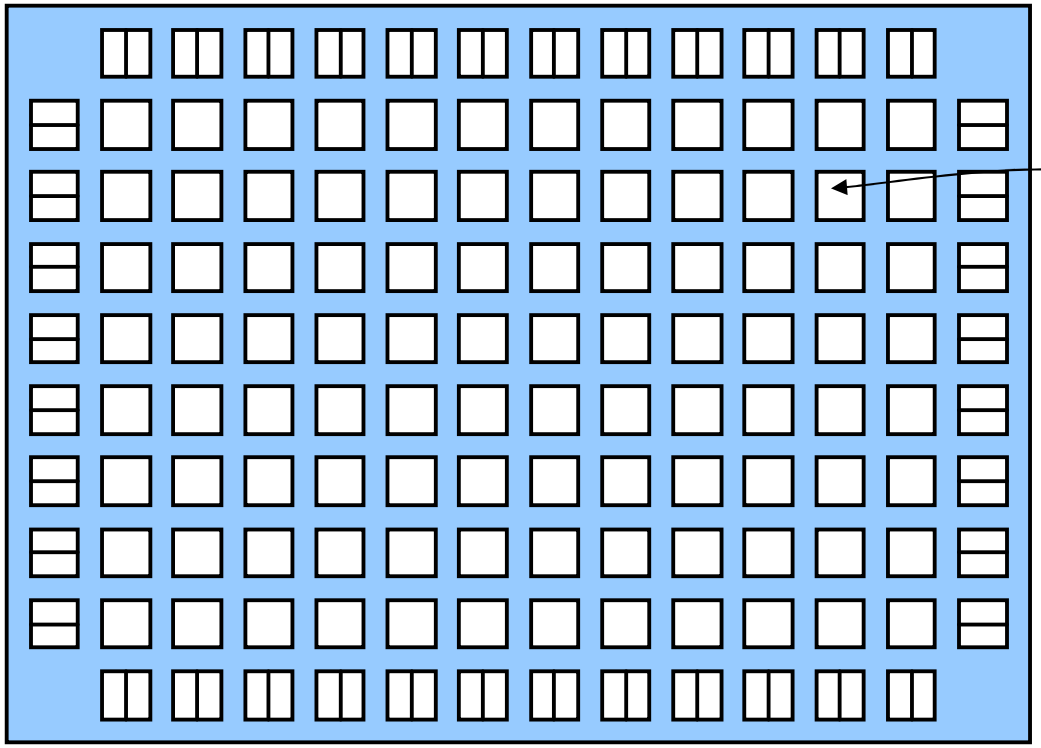



FPGA Die Photograph (not likely a Cyclone V)

Intel Cyclone V FPGA Die Floorplan



What's Inside an FPGA?



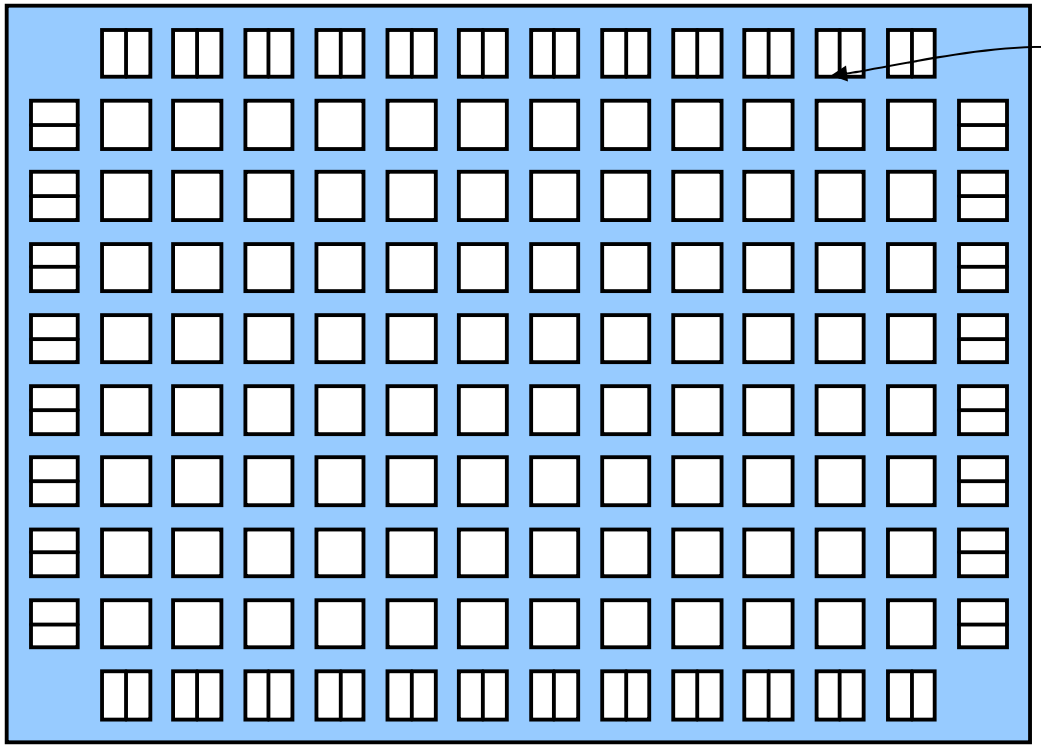
Logic Blocks

- used to implement logic
- lookup tables and flip-flops

Altera: LABs

Xilinx: CLBs

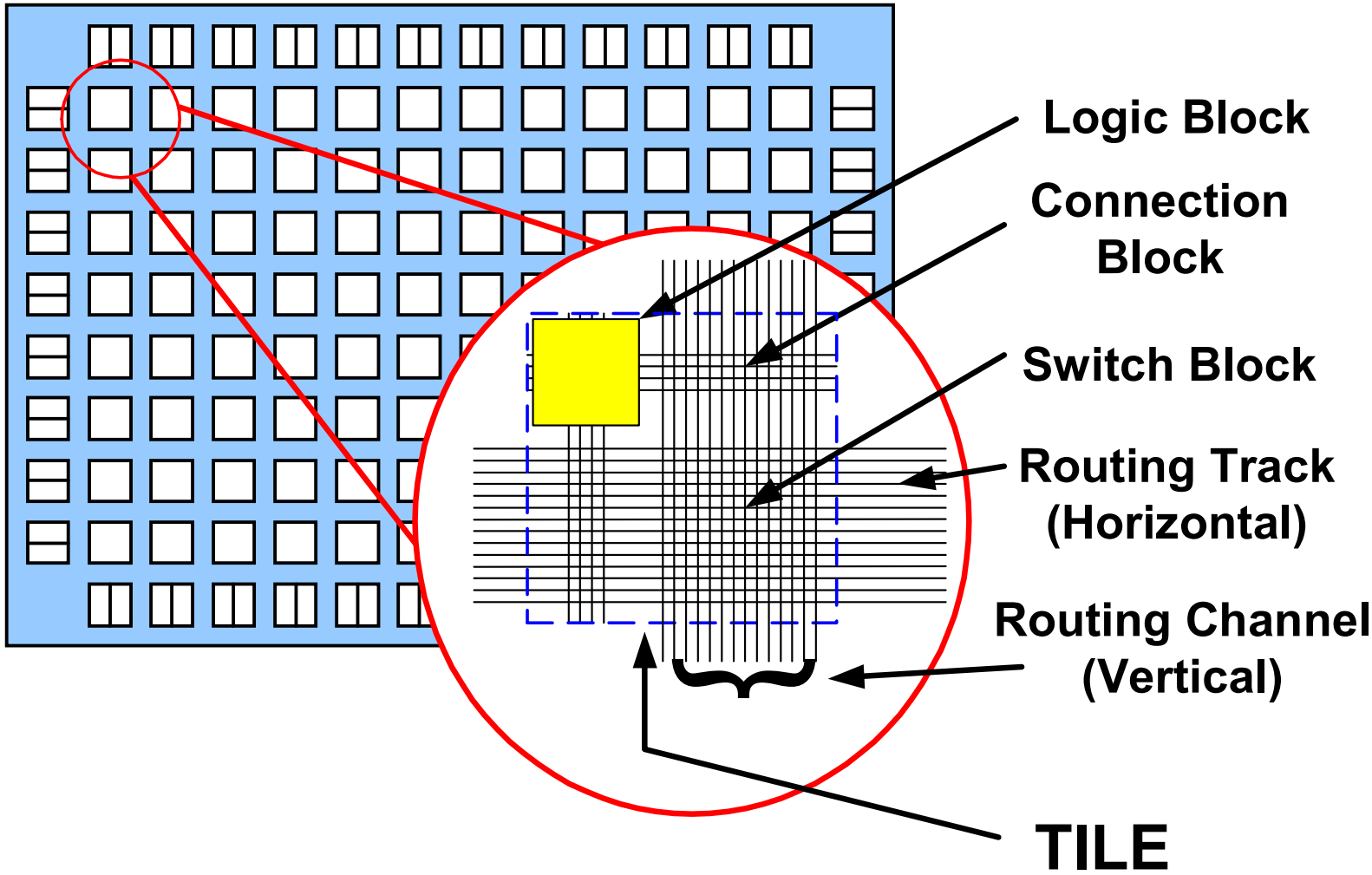
What's Inside an FPGA?



I/O Blocks

- interface off-chip
- can usually support many I/O Standards

What's Inside an FPGA?



Synthesis

Synthesis converts your Verilog code to a collection of logic blocks connected using logical wires.

Two aspects:

1. Map behaviour to logic equations (gates) ← **For FPGAs or custom chips**
2. Map gates to FPGA logic blocks ← **FPGA-Specific**

We will talk about each of these.

Synthesis: Step 1: Converting HDL to Gates

Each process/always in your design is compiled independently:

```
process (all)
begin
    d <= a and b and (not c);
end process;
```

```
process (clk)
begin
    if clk='1' then
        a <= f or g;
    end if;
end process;
```

```
b <= h and i;
```

```
z <= h or i;
```

```
process (all)
begin
    f <= d or e;
end process;
```

(don't worry about what this code does, it is just meant as an illustration of a design with multiple processes).

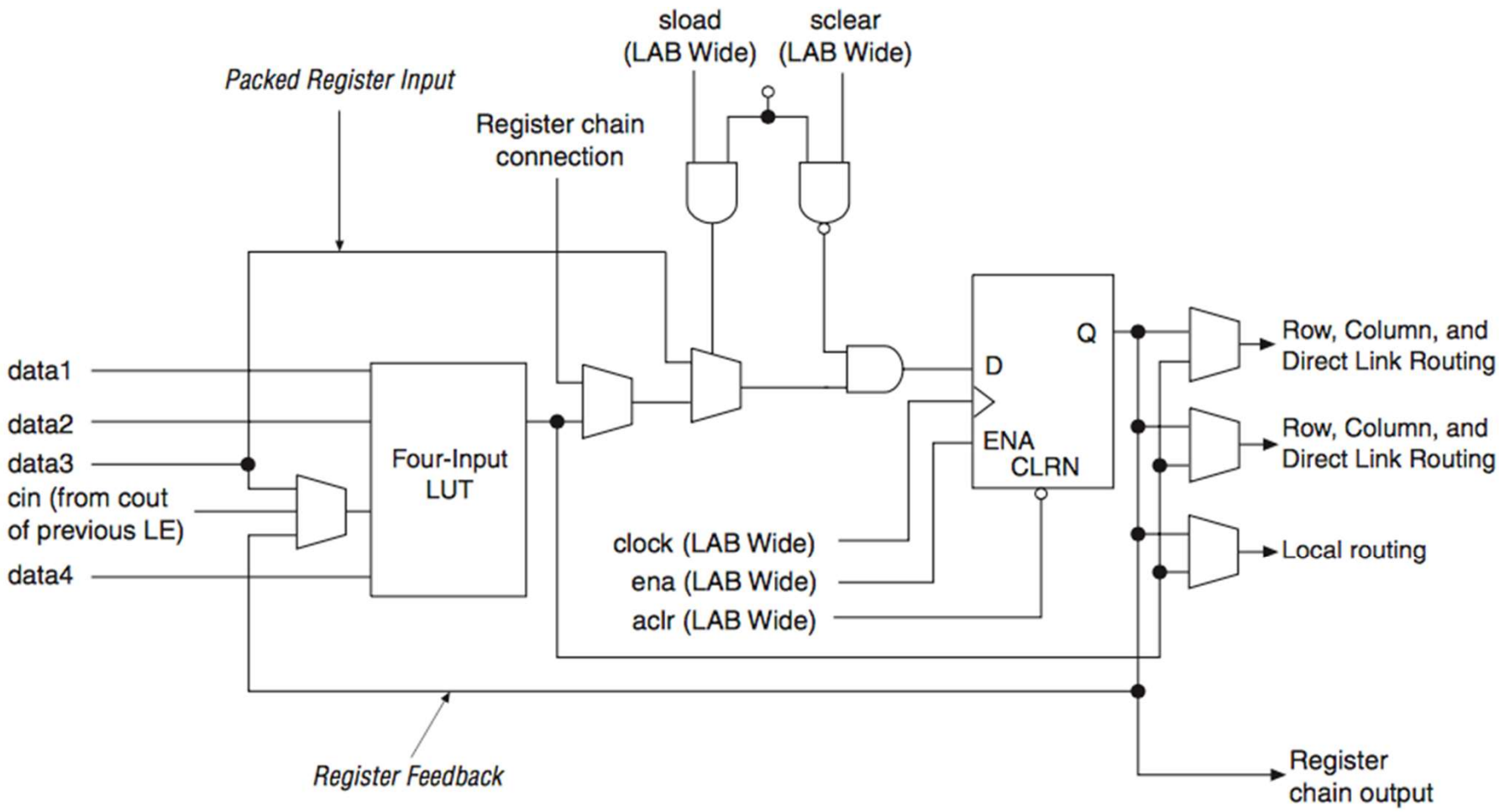
When you compile to hardware, <u>each of these units is compiled to a separate hardware unit</u>, and they operate in parallel.

All <<Search>>

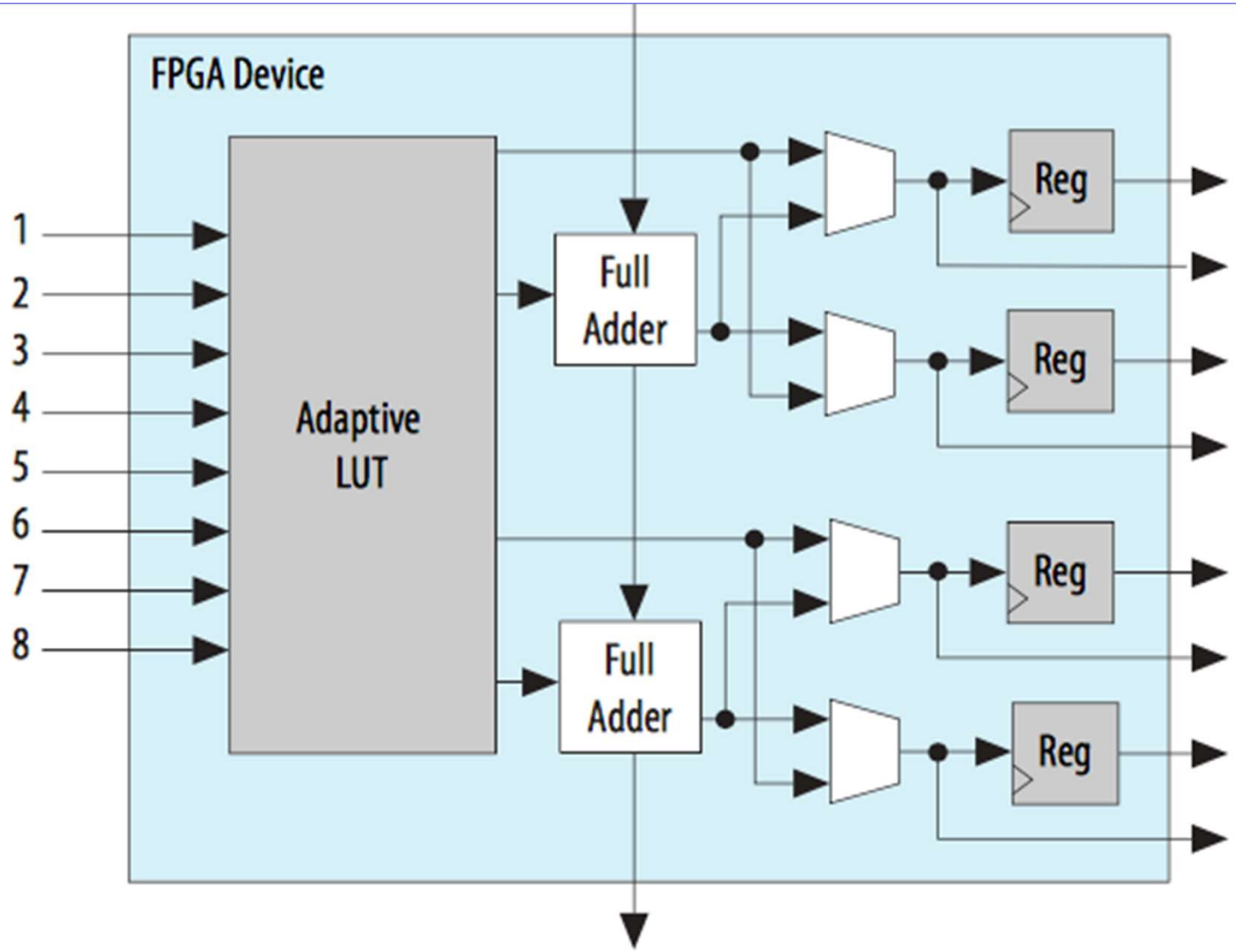
Type	ID	Message
	12021	Found 1 design units, including 1 entities, in source file db/add_sub_npi.tdf
	12130	Elaborated megafunction instantiation "lpm_add_sub:Add12"
	12133	Instantiated megafunction "lpm_add_sub:Add12" with the following parameter:
	12021	Found 1 design units, including 1 entities, in source file db/add_sub_opi.tdf
	17049	121 registers lost all their fanouts during netlist optimizations.
	16010	Generating hard_block partition "hard_block:auto_generated_inst"
	21074	Design contains 16 input pin(s) that do not drive logic
	21057	Implemented 1052 device resources after synthesis - the final resource count might be different
		Quartus II 32-bit Analysis & Synthesis was successful. 0 errors, 72 warnings
		**********
		Running Quartus II 32-bit Fitter

Logic Blocks implement the functionality of the circuit

Cyclone II logic block (on the DE2 board)

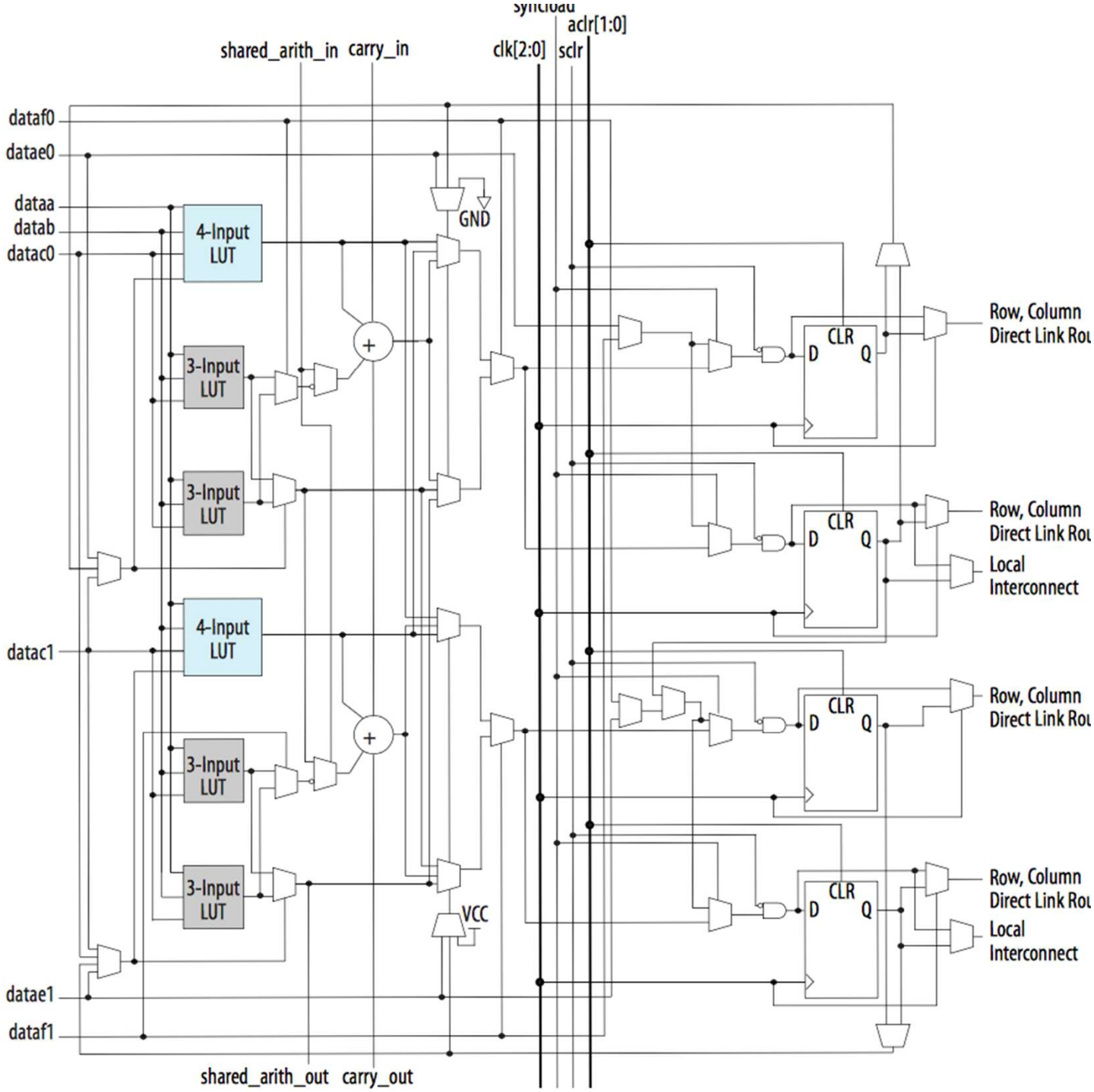


Cyclone V logic block (on the DE1-SoC / DE0-CV)



From Altera, Cyclone V Overview, CV-51001, June 2016.

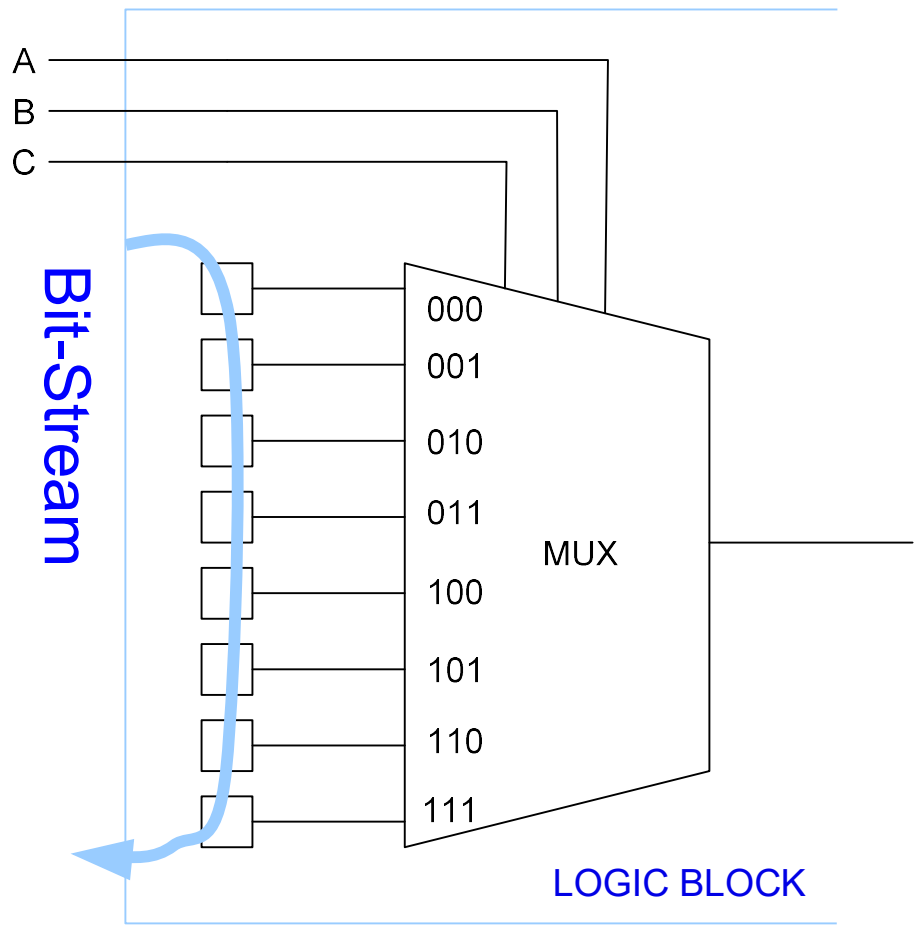
ALM Consists of a number of LUTs:



From Altera, Cyclone V Handbook, CV-52001, Jan 2015.

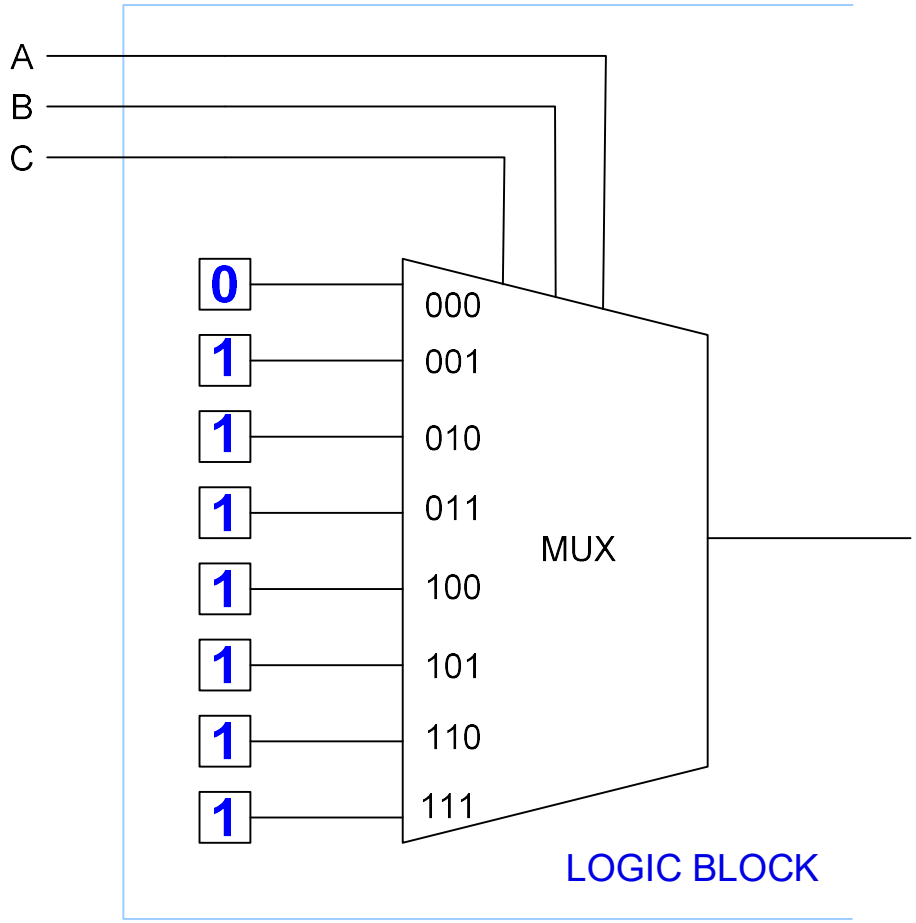
Key item in both is a look-up table:

Basic Logic Gate: Lookup-Table (LUT)



Function of each lookup table can be configured by shifting in bit-stream.

Quick Question: What would this function implement?



A LUT is simply a way of building the truth table in silicon

C	B	A	F
0	0	0	**0**
0	0	1	**1**
0	1	0	**1**
0	1	1	**1**
1	0	0	**1**
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

$$F = A + B + C$$

Interesting and useful fact: A LUT with K inputs can implement any function with up to K inputs

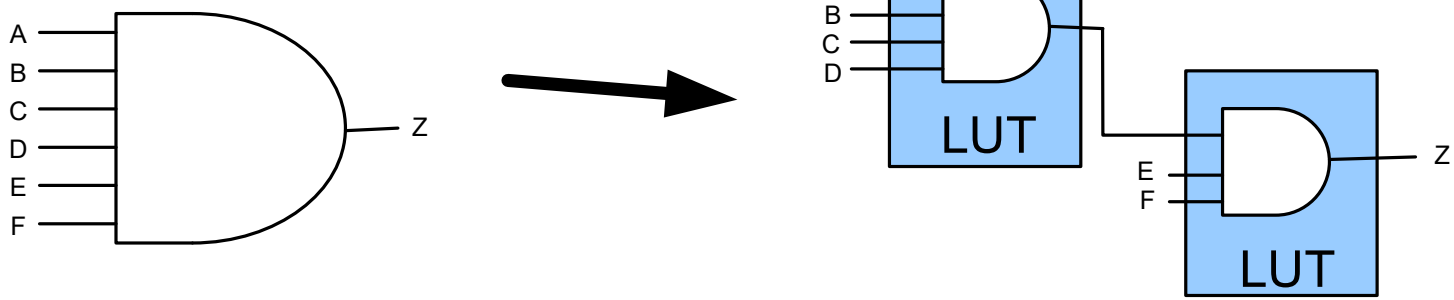
So a 4-input LUT can implement a 4-input AND, or a 3-input NOR, or actually any function with 4 or fewer inputs.

Implementing Circuits with Logic Blocks

Suppose we are using an FPGA containing 4-input LUTs.

- We need to break up our circuit into “chunks”, where each chunk has at most 4 inputs
- Each “chunk” can be implemented by one LUT

Logic Circuit

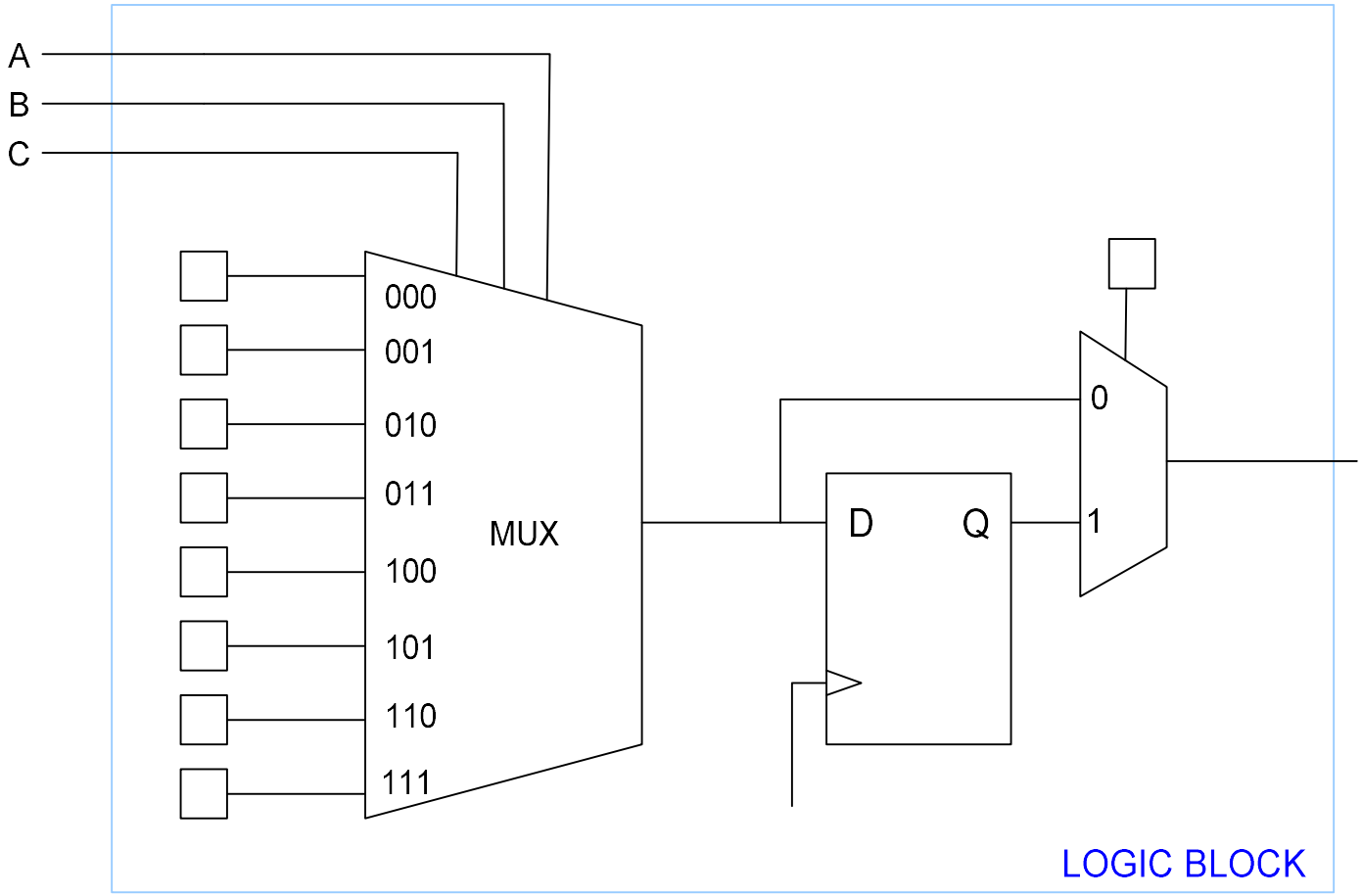


Implementation on an FPGA

Note: This is done automatically by CAD tools (Quartus II)

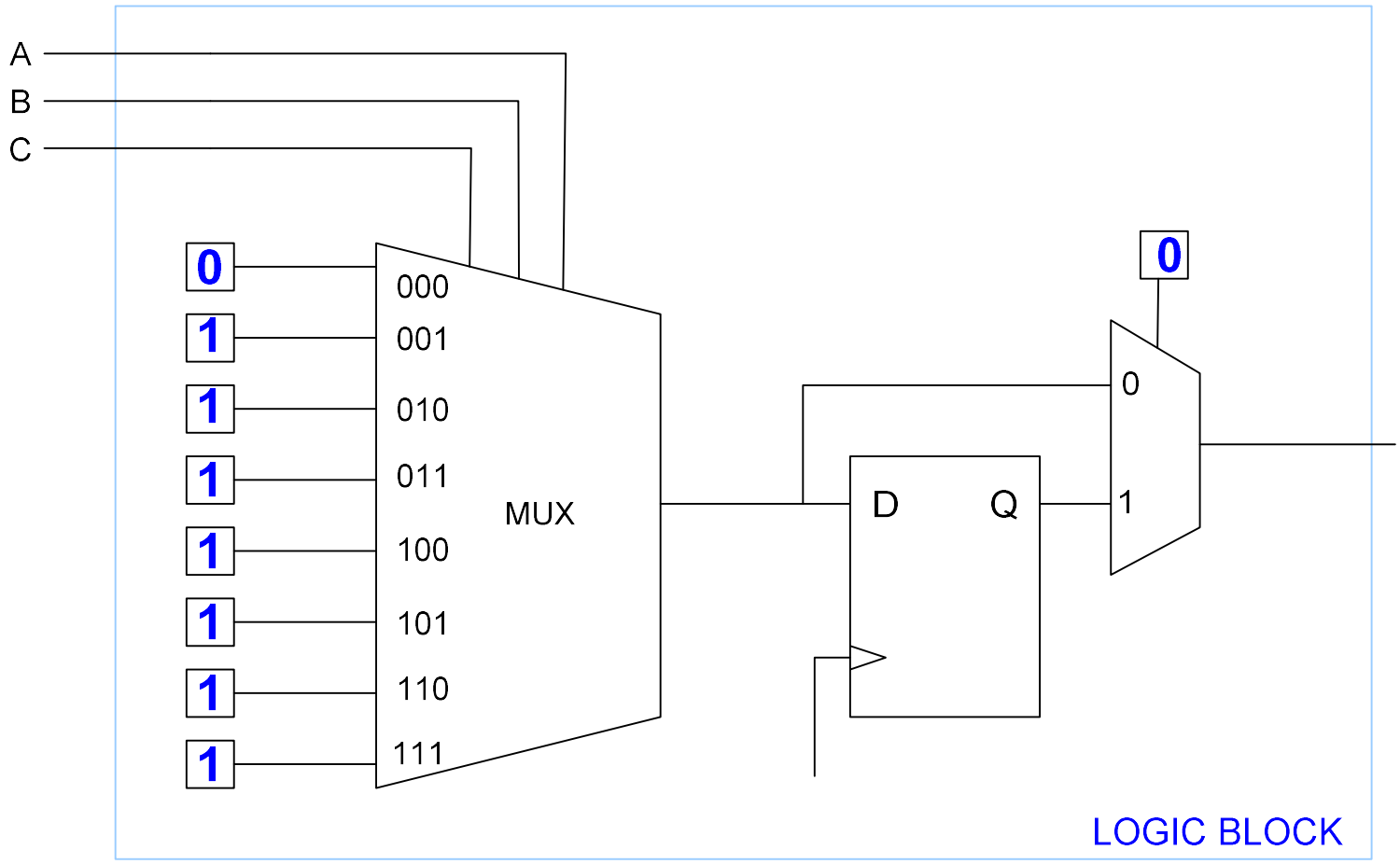
Logic Block:

Usually a logic block also contains a flip-flop

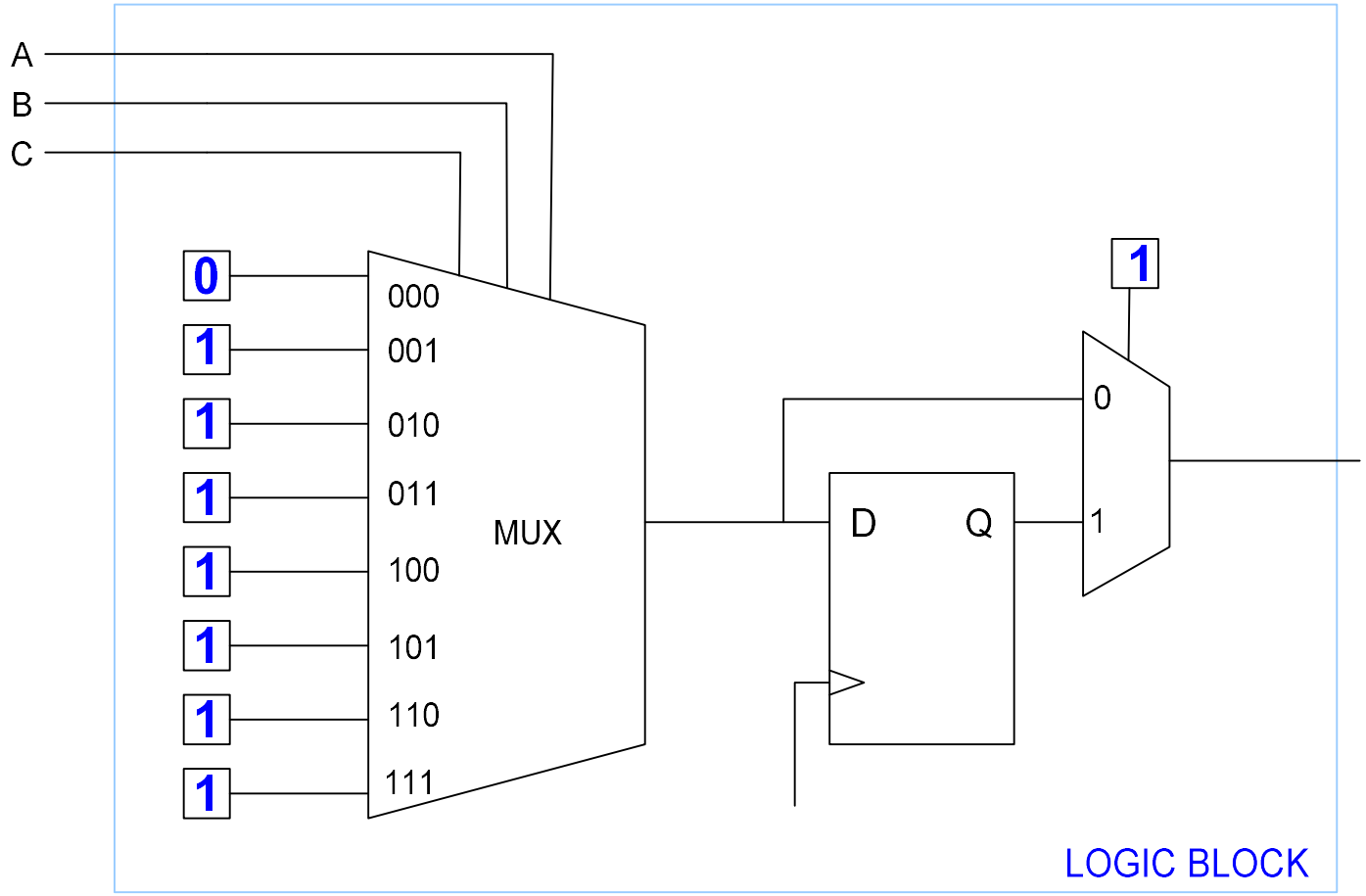


This logic block can implement combinational and sequential functions.

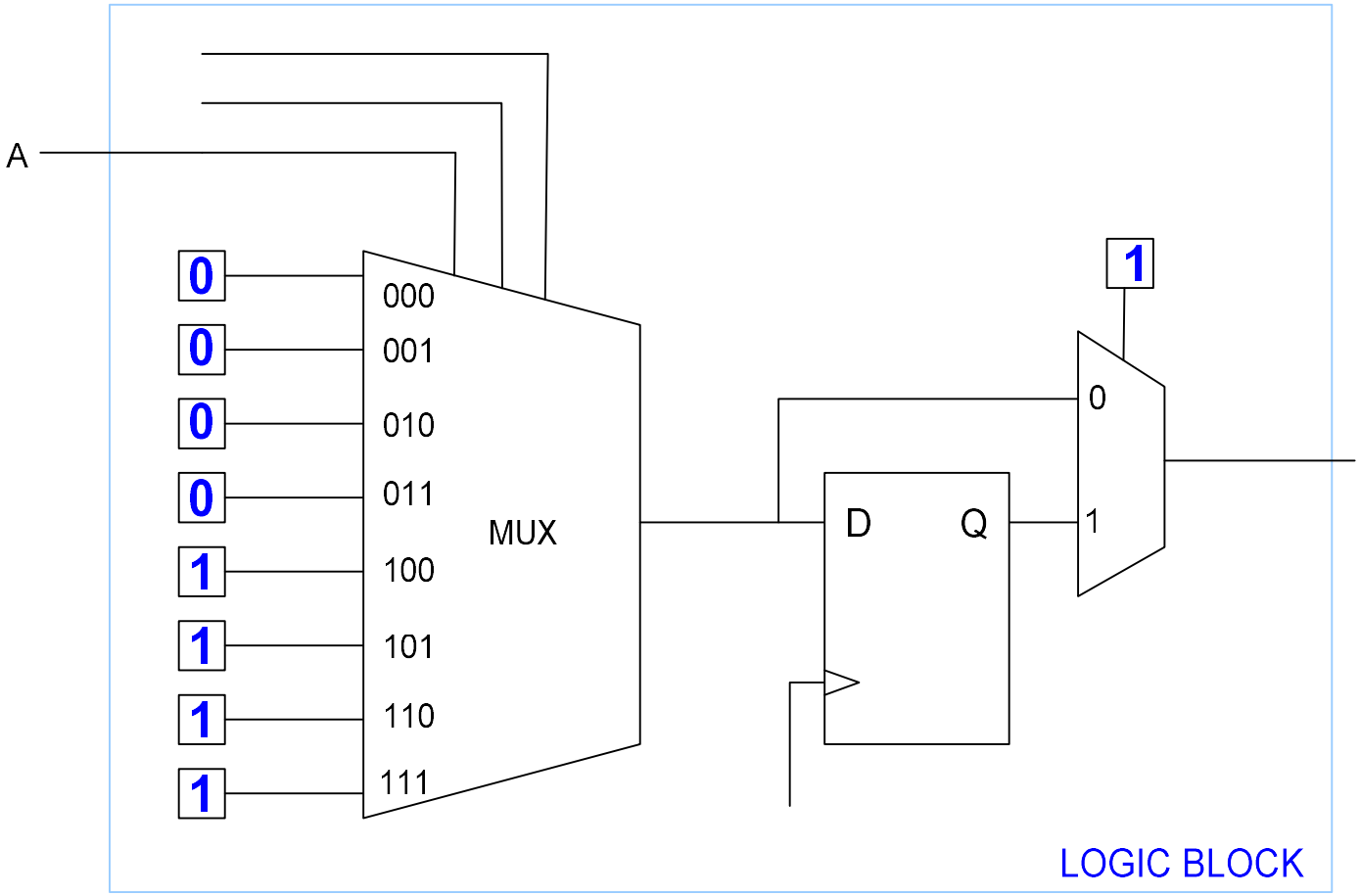
Example: This implements A+B+C as before



Example: This implements A+B+C *followed by a flip-flop*



Suppose we want to implement a flip-flop only?



Logic Block:

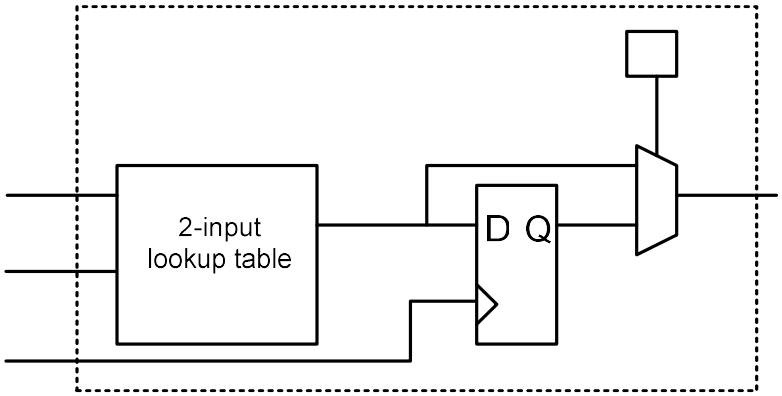
So each of these logic blocks can implement one of:

- A combinational function
- A combinational function followed by one flip-flop

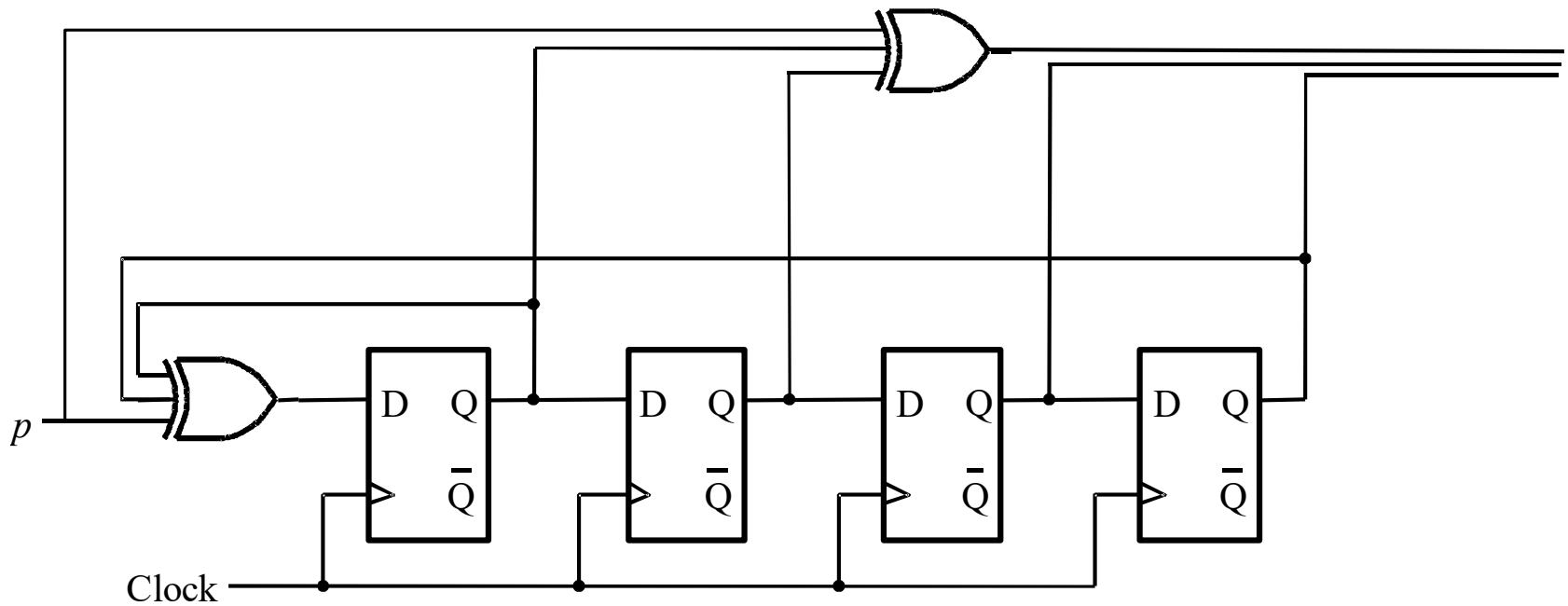
If the combinational function is “unity”, then the logic block can be used to implement just a flip-flop

This means if a circuit requires N flip-flops, it needs *at least* N logic blocks.

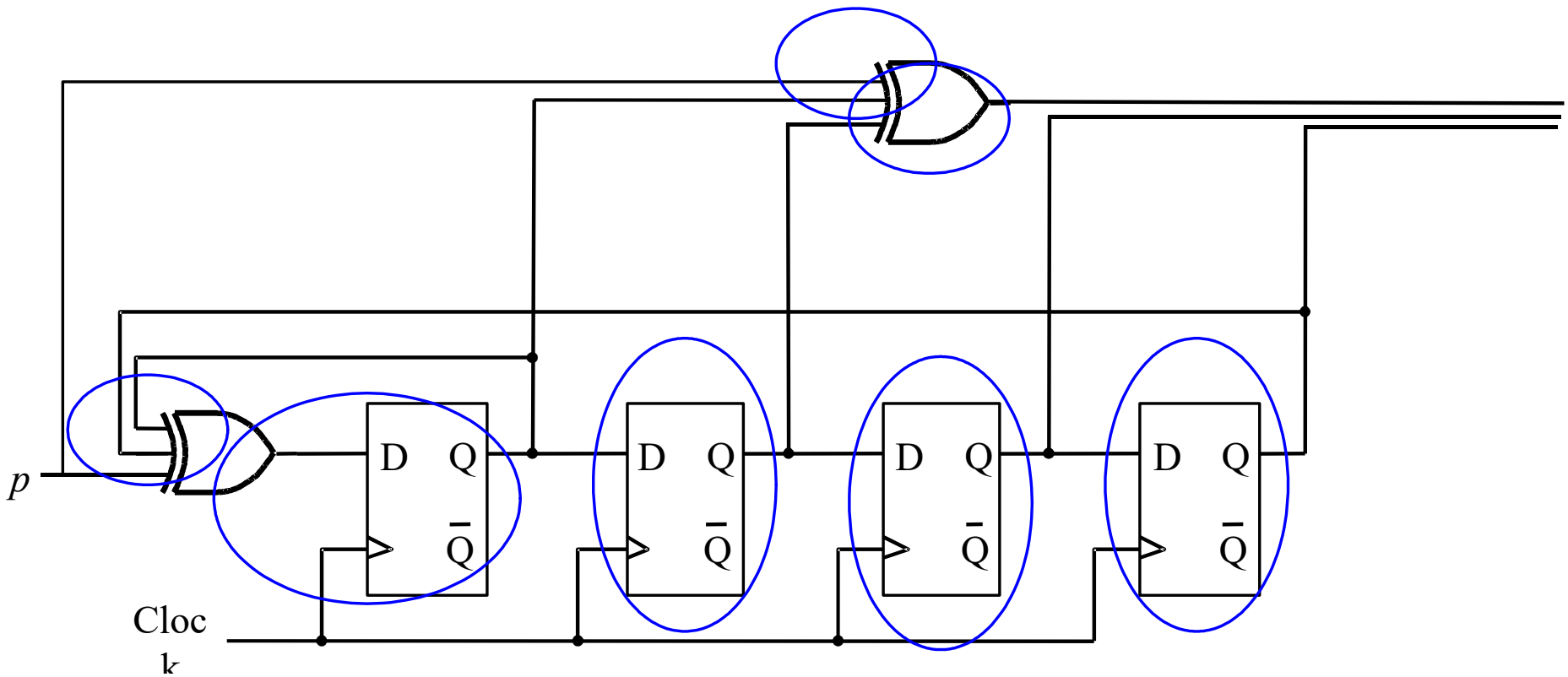
Another example. Say we have an FPGA with this logic block



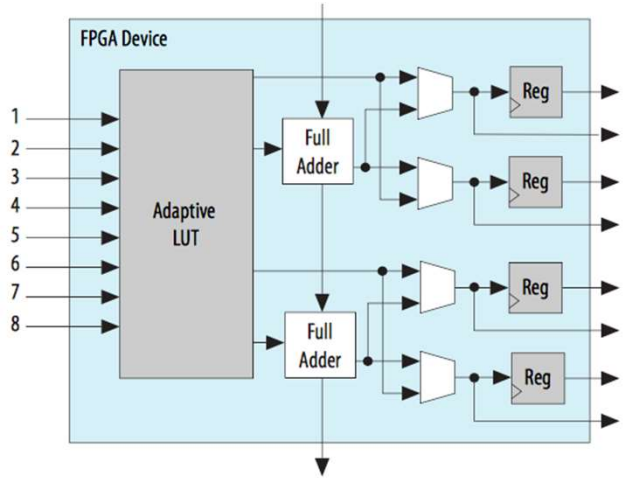
And we want to implement this circuit:



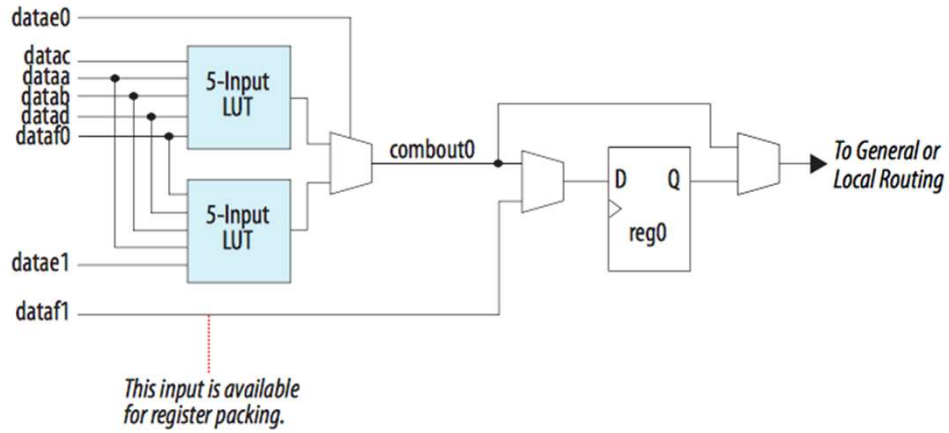
We need 7 logic blocks:

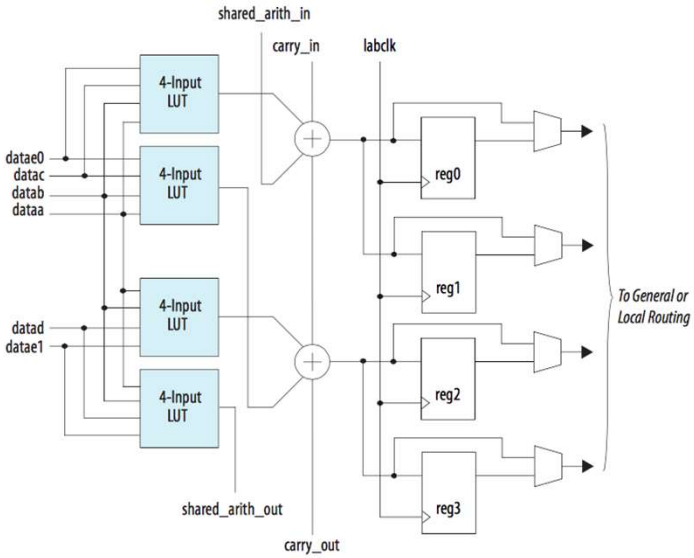


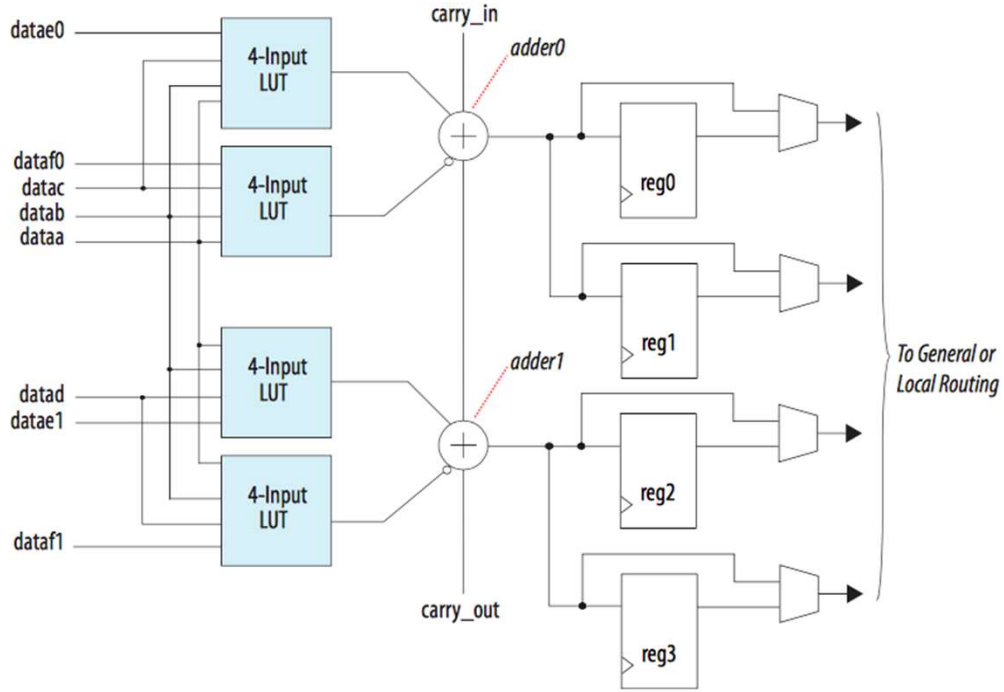
The ALM on the Cyclone V is “fracturable”:



Can be configured as one of:







How many logic blocks in a typical FPGA?

The Cyclone II on the DE2 has 33,216 LEs (logic elements).

The Cyclone V on the DE1 has 18,480 ALMs (but they are bigger)

Higher end FPGAs have more: Xilinx Virtex-7:

Part	Logic Cells
XC7VX330T	326,400
XC7VX415T	412,160
XC7VX485T	485,760
XC7VX550T	554,240

Part	Logic Cells
XC7VX690T	693,120
XC7VX980T	979,200
XC7VX1140T	1,139,200

(definition of Logic cell is a bit different for Xilinx parts, but the point is the same)

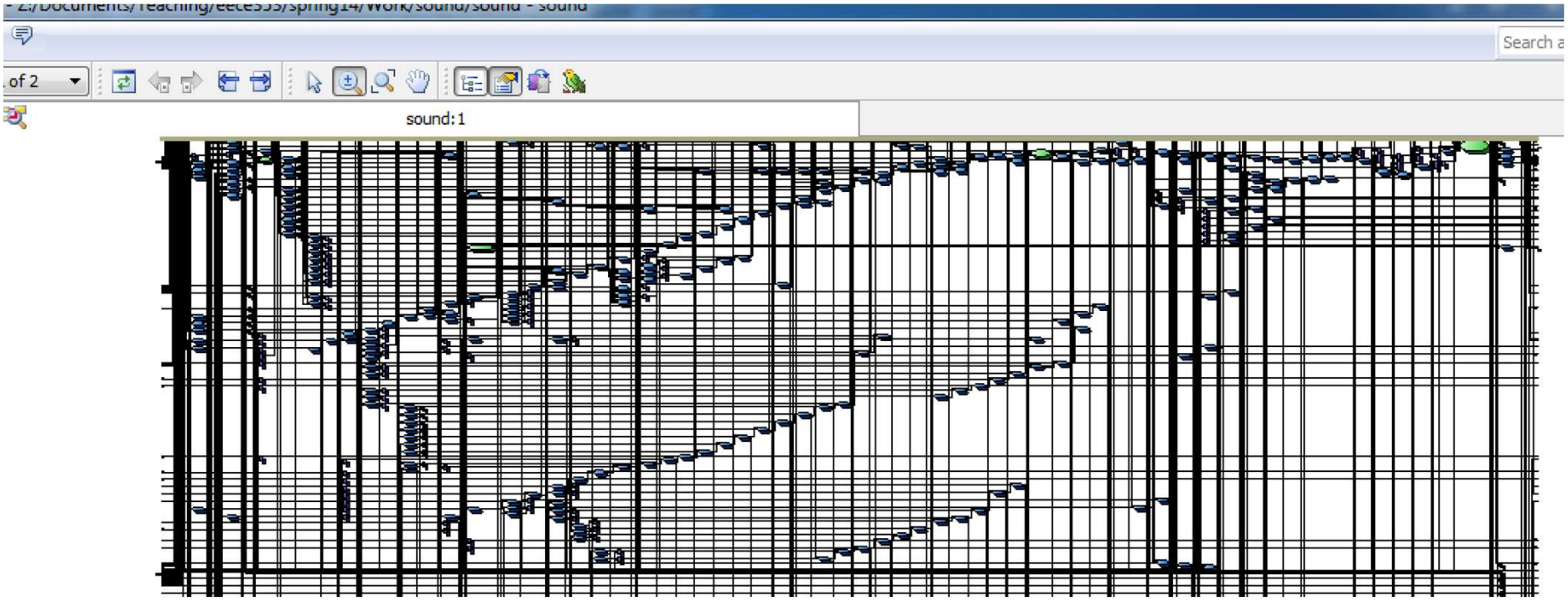
You can see how much you use in the report:

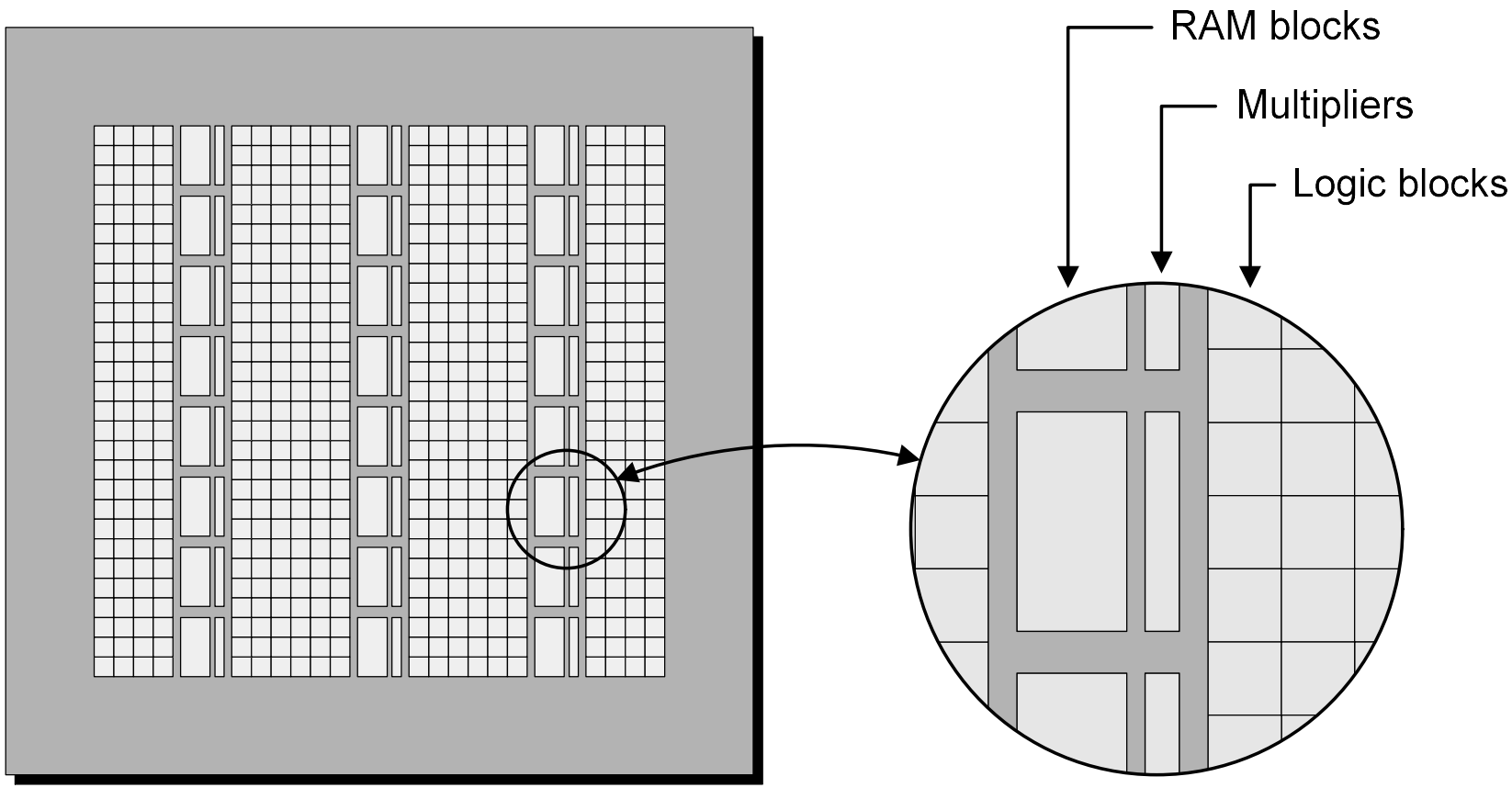
Flow Summary	
Flow Status	Successful - Thu Mar 20 16:45:
Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP
Revision Name	sound
Top-level Entity Name	sound
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	956 / 33,216 (3 %)
Total combinational functions	956 / 33,216 (3 %)
Dedicated logic registers	418 / 33,216 (1 %)
Total registers	418
Total pins	34 / 475 (7 %)
Total virtual pins	0
Total memory bits	6,144 / 483,840 (1 %)

956 Logic Elements

For Cyclone V, results are reported in terms of ALMs (Adaptive Logic Modules, each with roughly two logic elements)

Choose Tools->Netlist Viewers->Technology Map Viewer




RAM blocks
Multipliers
Logic blocks

Embedded Memory Blocks:

Stratix 10:

Each embedded memory block has 20Kbits

Device	M20K Blocks	Total Memory Bits
SX500	2196	43M
SX650	2583	50M
SX850	3477	68M
SX1100	4401	86M
SX1650	5851	144M
SX2100	6501	127M
SX2800	11721	229M

Configuration (# words x # bits per word)
2K (words) x 10 (bit)
1K x 20
512 x 40

Each M20K memory block can be configured in one of three configurations.

Other FPGAs have similar memory structures.

Embedded Memory Blocks:

Altera Cyclone V SE (DE1-SoC board) and V E (DE0-SoC board)

Each embedded memory block has 10Kbits

Device	M10K Blocks	Total Memory Bits
5CSEMA2	140	1,433,600
5CSEMA4	270	2,764,800
5CSEMA5	397	4,065,280
5CSEMA6	557	5,703,680

Device	M10K Blocks	Total Memory Bits
5CEBA2	176	1,802,240
5CEBA4	308	3,153,920
5CEBA5	446	4,567,040
5CEBA7	686	7,024,640
5CEBA9	1220	12,492,800

Cyclone II/V Memory Blocks

Each memory block can be configured in one of several modes:

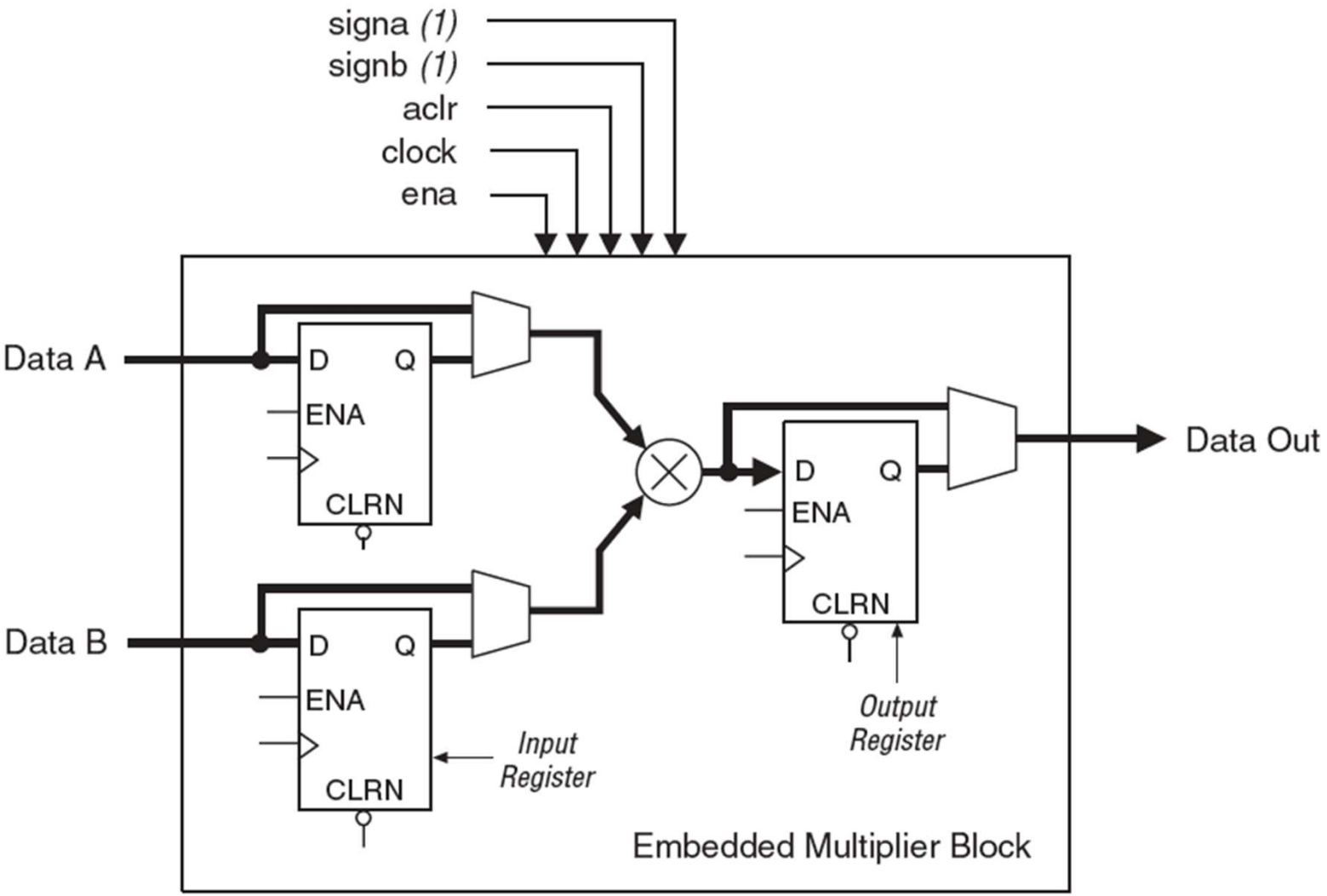
Cyclone II

Configuration (# words x # bits per word)
4K (words) x 1 (bit)
2K x 2
1K x 4
512 x 8
512 x 9
256 x 16
256 x 18
128 x 32
128 x 36

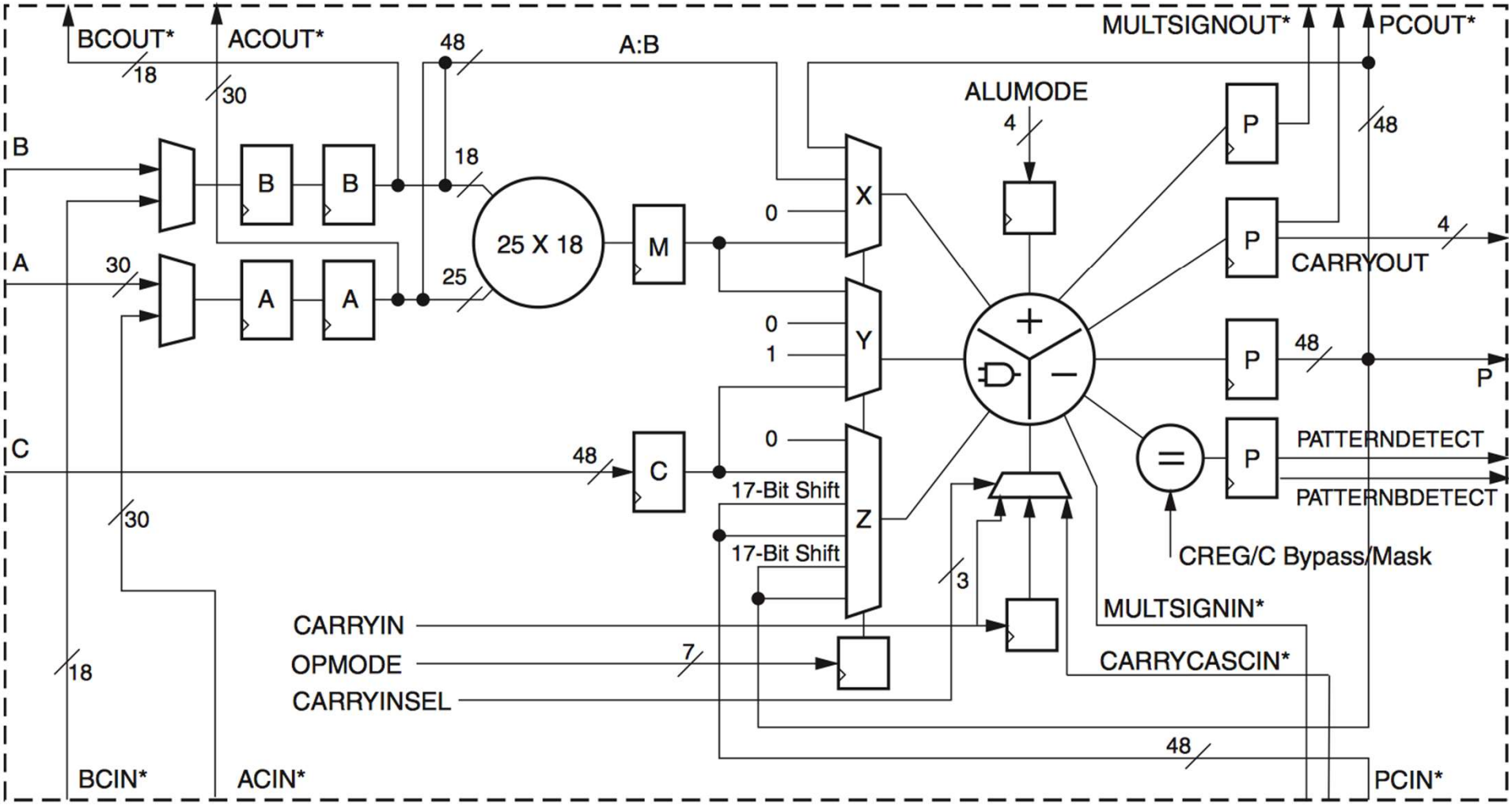
Cyclone V

Configuration (# words x # bits per word)
8K (words) x 1 (bit)
4K x 2
2K x 5
2K x 4
1K x 8
1K x 10
512 x 16
512 x 20
256 x 32
256 x 40

Multiplier Block Architecture:

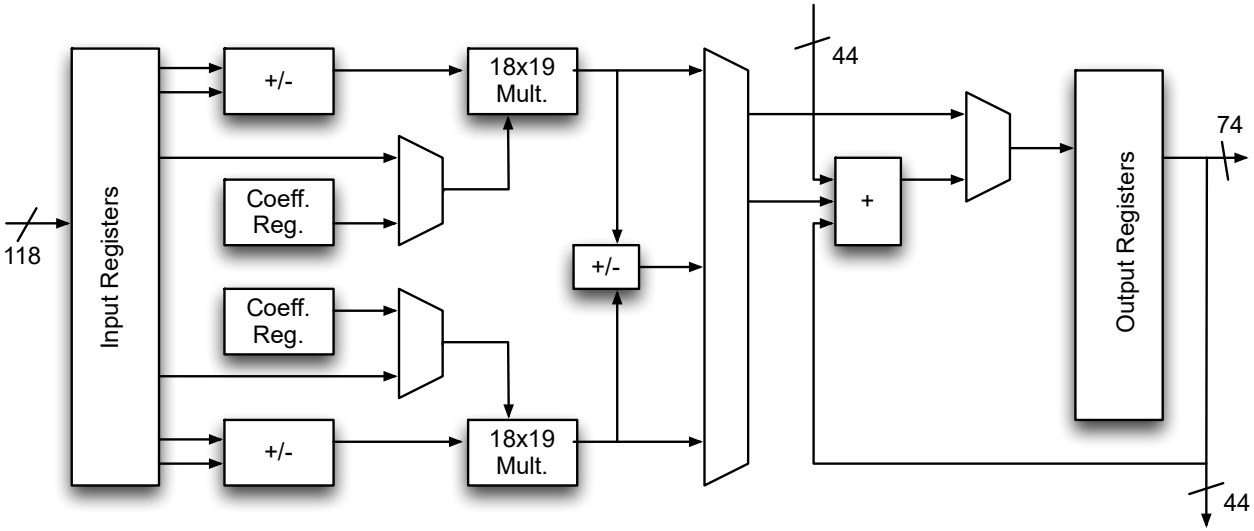


Xilinx DSP48E (Virtex 5 FPGA)



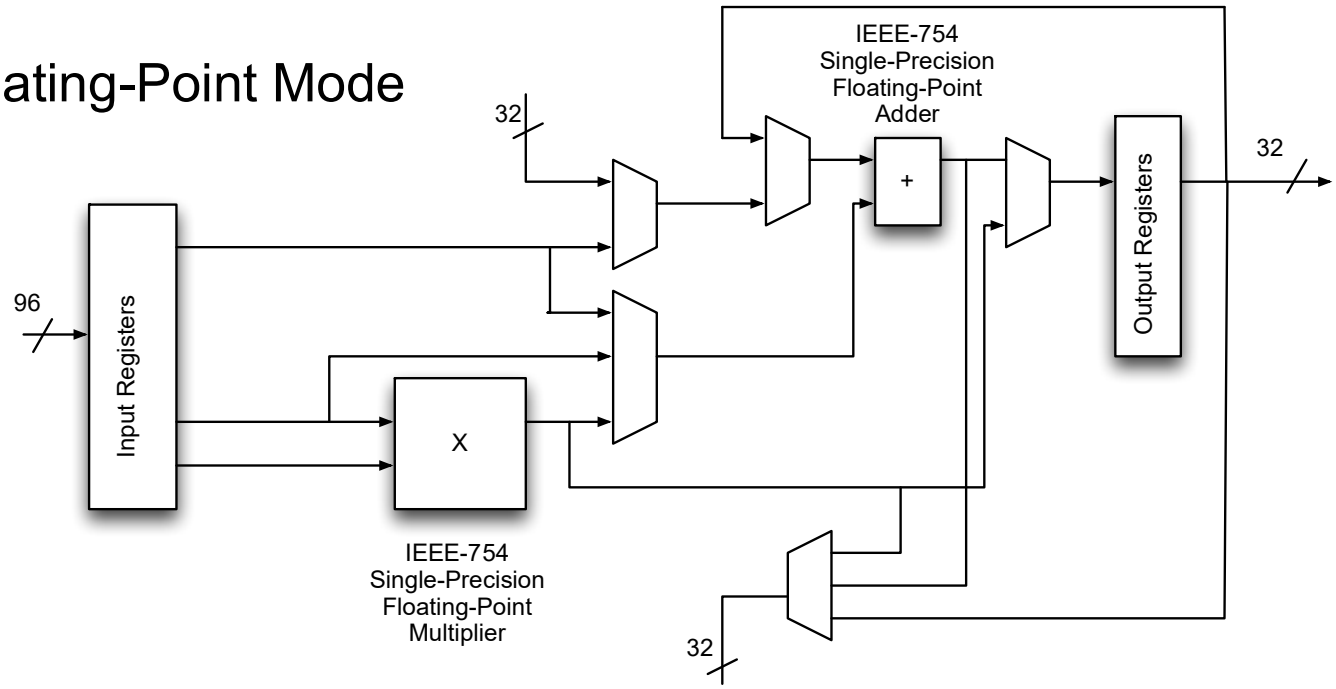
From Xilinx, inc, "Virtex-5 FPGA XtremeDSP Design Considerations."

DSP Architecture (Stratix 10):



Fixed-Point Mode

Floating-Point Mode



Multipliers:

Altera Cyclone II:

Device	Embedded Multipliers	9x9 Multipliers	18x18 Multipliers
EP2C5	13	26	13
EP2C8	18	36	18
EP2C20	26	52	26
EP2C35	35	70	35
EP2C50	86	172	86
EP2C70	150	300	150

DSP Blocks:

Altera Stratix 10:

Device	Embedded DSP Blocks
SX500	1152
SX650	1440
SX850	2016
SX1100	2520
SX1650	3145
SX2100	3744

Other FPGAs are similar. These DSP blocks provide extremely fast and efficient implementations of DSP-type applications.

Inferring Multiplications:

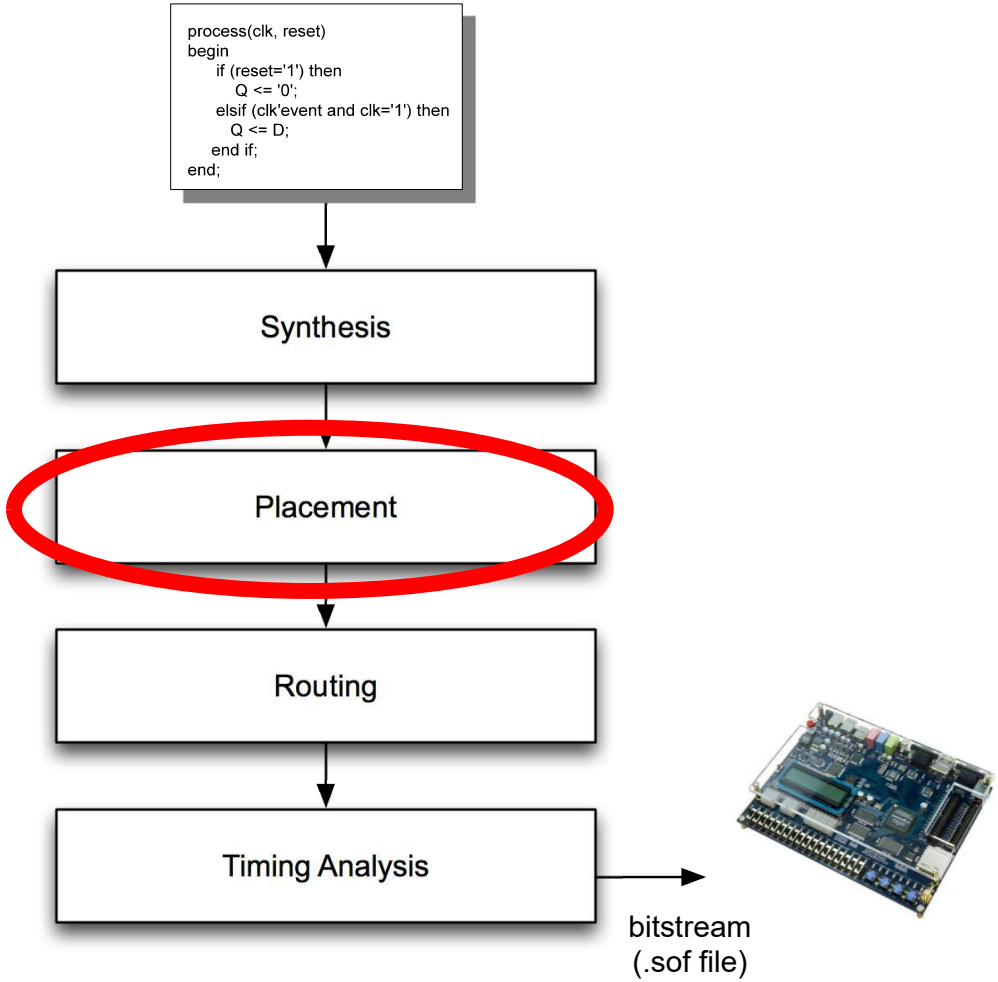
```
wire [7:0] a;
wire ]7:0] b;
wire [15:0] c;
```

```
assign c = a * b;
```

Dedicated logic registers	18 / 33,216 (< 1 %)
Total registers	18
Total pins	79 / 322 (25 %)
Total virtual pins	0
Total memory bits	131,072 / 483,840 (27 %)
Embedded Multiplier 9-bit elements	1 / 70 (1 %)
Total PLLs	0 / 4 (0 %)

FPGA Compilation:

Next step is placement, in which the tool finds a physical location for each logic block in the circuit.



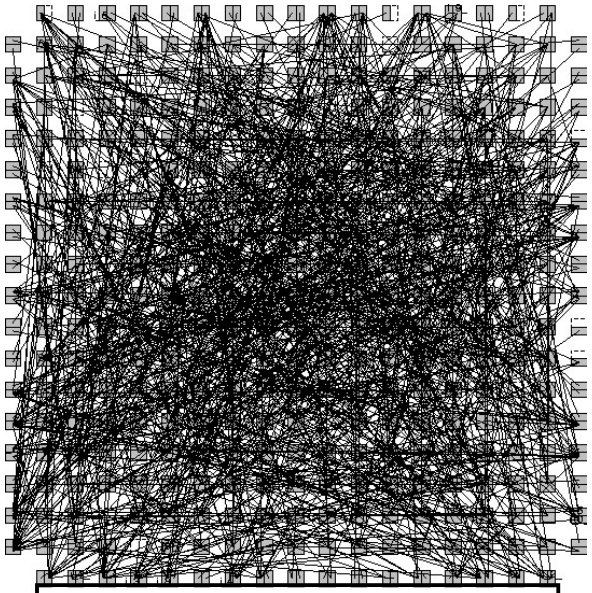
All 16 <<Search>>

Type	ID	Message
⚠	15058	PLL "clock_generator:my_clock_gen|altpll:DE_Clock_Generator_Audio|pll" is in normal or sou
⚠	15064	PLL "clock_generator:my_clock_gen|altpll:DE_Clock_Generator_Audio|pll" output port clk[0]
⚠	15705	Ignored locations or region assignments to the following nodes
ℹ	171121	Fitter preparation operations ending: elapsed time is 00:00:01
ℹ	170189	Fitter placement preparation operations beginning
ℹ	170190	Fitter placement preparation operations ending: elapsed time is 00:00:01
ℹ	170191	Fitter placement operations beginning
ℹ	170137	Fitter placement was successful
ℹ	170192	Fitter placement operations ending: elapsed time is 00:00:01
ℹ	170193	Fitter routing operations beginning
ℹ	170195	Router estimated average interconnect usage is 0% of the available device resources

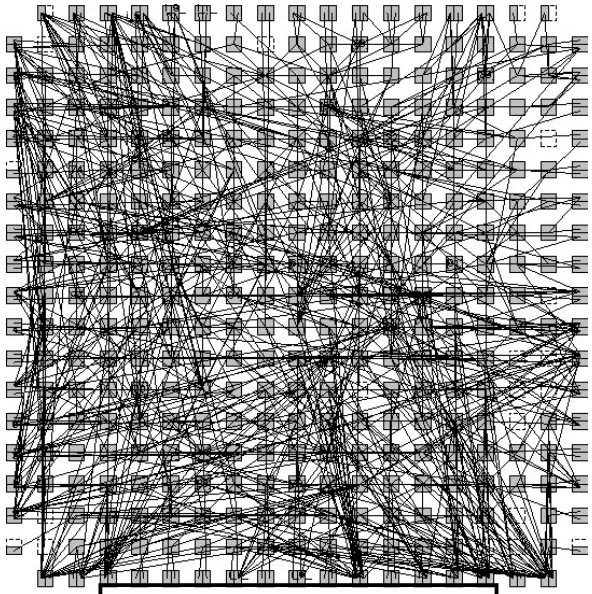
Placement

Goal of placement:

1. Place logic blocks that are connected close to each other
2. Put extra emphasis on shortening the critical path

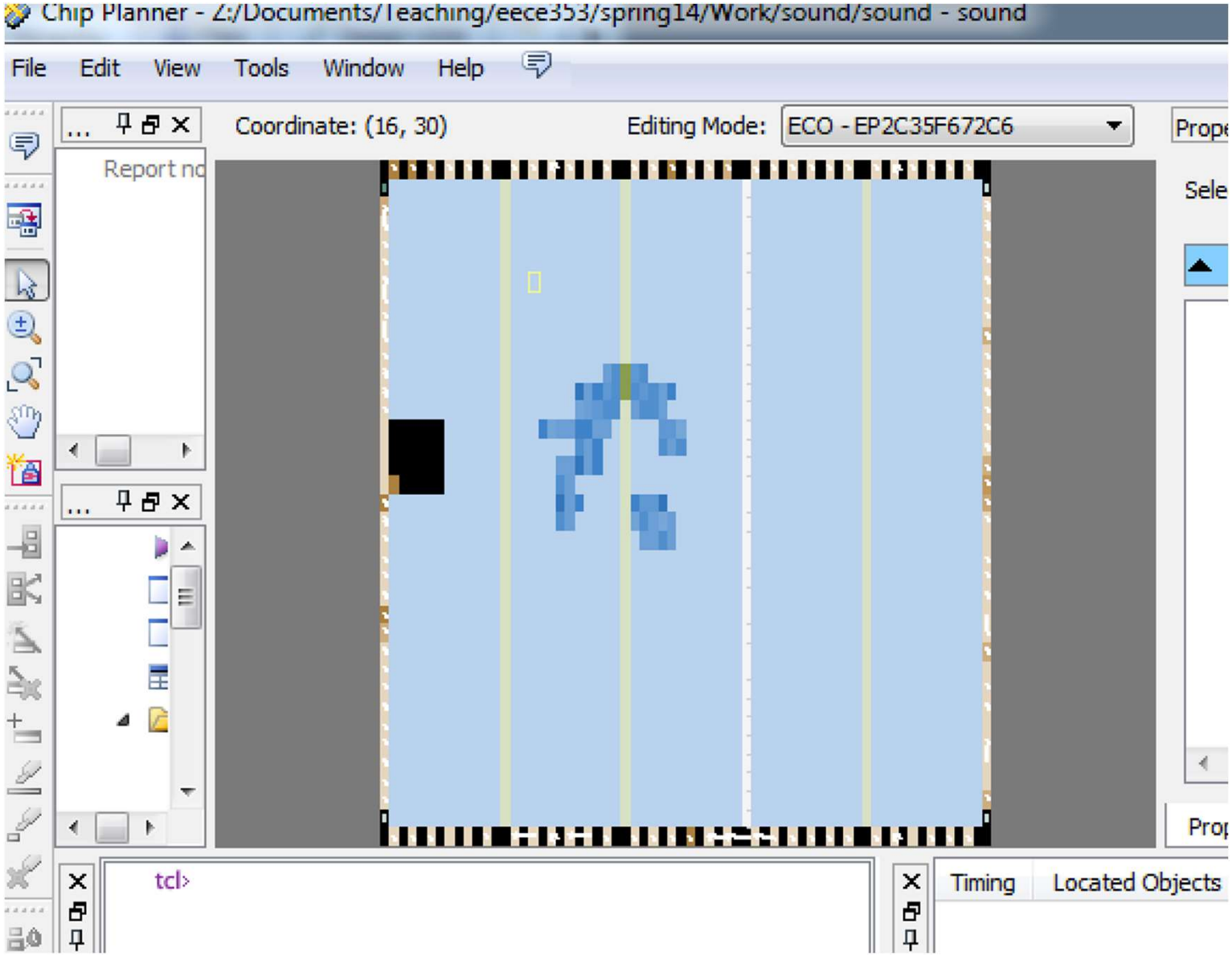


Random Initial Placement



Final Placement

Choose Tools->Chip Planner

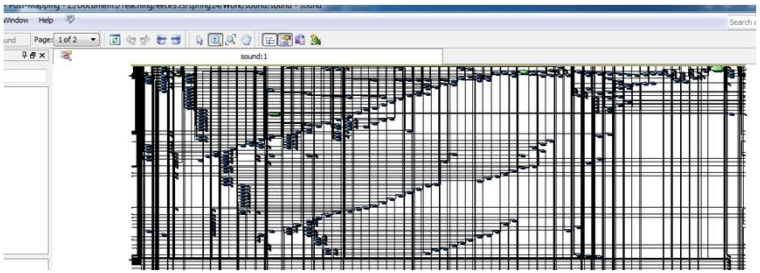


```
Type   ID      Message
       15058   PLL "clock_generator:my_clock_gen|altpll:DE_Clock_Generator_Audio|pll" is in normal or sou
       15064   PLL "clock_generator:my_clock_gen|altpll:DE_Clock_Generator_Audio|pll" output port clk[0]
       15705   Ignored locations or region assignments to the following nodes
      171121   Fitter preparation operations ending: elapsed time is 00:00:01
      170189   Fitter placement preparation operations beginning
      170190   Fitter placement preparation operations ending: elapsed time is 00:00:01
      170191   Fitter placement operations beginning
      170137   Fitter placement was successful
      170192   Fitter placement operations ending: elapsed time is 00:00:01
      170193   Fitter routing operations beginning
      170195   Router estimated average interconnect usage is 0% of the available device resources
```

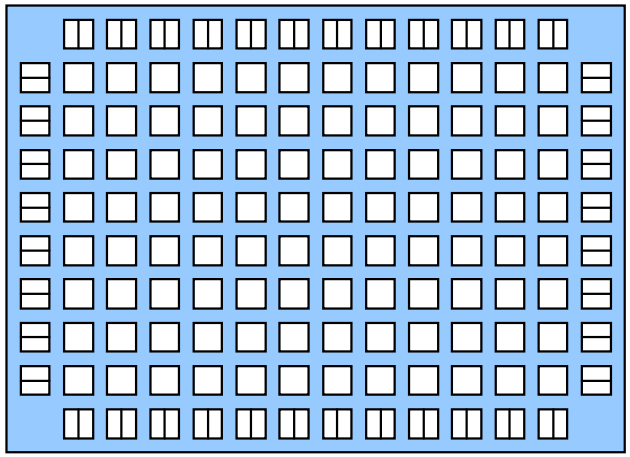
In this case, very fast. But in general, placement could take hours. Why?

There are many different placement solutions, and we would like to choose the best -> leads to fastest clock speed.

From a previous CPEN 311 lab:



956 Circuit Elements to Place



33,216 Logic Blocks on the device

How many different ways are there to place the 956 Circuit Elements on the FPGA?

33216 * 33215 * 33214 * … * 32262 * 32261 = ?

=

724588565015760387803217458682543740244038597713763730680406644977633590429572812614493219737577863360840616993376759400138230336130381203225920672817377540878869761386595657272490708218529023039090932235948335629979597545343611907113249295685309331418837576928044232286869852793480179326283152657477395440801999289318498305092690008812091751761257534883005620445865736634403144399642359172224824244685888326615671446136252974662622532690473937798773340107466877619816142823163880134156176842252662781798124254079939302404022673781478631664765504023888378070571251393979798506224575496795578717372530262028120548056827464085672902267822152262736280617103762709858919955110924561816593059840105938521387914893164853043581583938990593772812391748662584162933100843523502337355657522615175904694526338193899218767916076621417411263780688861030416525704872528781604850560122480062154356925773450968993238518493713756858248454150258788630733472345949561685999190783194216571597948443556250271727528550907221561399231166189359210706496399753305636697416272009662627367118159012115668176174818229722325664821751936773406794135368399465597963840572628391441811414294513507281655623736018473741467514412227223354661559109959296030191307751027826501754894829247471955580053646053695874638149302028840997678926471637465949923432163469742556961927653796761191065319322710816796945919821747164028846736462661603228630703588507961207160623358697569775405073870940061217700482322607113481715412182320019563843198834248149540525870662535451317949887839587763131885405464544530630232773304755607340261537194018105991797486016340252212848568840022267280245156860654856659137005814986674470403586873926771187995335637535972179109412472681895523340191304303565833128373874886345241971864380729123108079700457572754064900736288524825839088367052299050834568894623541443487982447857107386778359291897662103837735213632231896369226041514955550622975946269583676051670172351416746395067926591426266365594084506193038976421623291733541483455610857274607126244210768312262952450004026249985642569061210574790038613662287541633838335803825069238522122801458260829823915132488001959168174845612486406826912900227920761169895858224362198301061988762137666745778079888691852552624149125297886031559900364818094045667231770012301592708068431224113687488514908455826573177043948310905779431070076184371303006704095325387724691599091078670136994877247095944515560888102590767832652095298227525401047409983971563646281174559848902733464617023960352217814336031769283544906014684270164161536452146604392180892205575927214496644494612141038824457516910643064275317948572874499048385266967947969074125219274081016979801086612602244599987419073320732180923043884902492086366861481272979548244451213212139626698556587499794004538346325001743181974359638229645529337709734321098291214782231922668383448235319079272153680586444575656325001818303406681898060077367695338862334644644249880046853114780490635876342882408261068440573742625709080826390013882088983819947473804505440690172028350530227919569930774959369095333356828872213445797820442343544273248112932618599150666234392587320812403051138258839455429350550859340631538482403187043920683683424240333110630285449223094897428527103939360856421478923348292366710794002917077625558062800483362894204665178138575004061365692439524815463174434353059243854937005060250458861604993277316893020955012963448407270170058140978063722487781023209317429815176401388238267090792307060783804277873688599662274926746125367067409048337423695290035683665850788660768806459348719731233195784443535511236606175307275919788750167969941857867274113088043457148709870687038318781702550476804048566779305553969821702661212718745073675419028189016861003738207433460327780564766283955999313843239223712162149531976859225840411473179831224936515938664397683600686007263384961429513634634582087338703442302386721529096633818653829011210024141720667437282407259423963717891004678170383110691962303104795273278905888030000555070165228917154313268183380968623794044225883179200105825697792000000000

Placement algorithms:

Naïve algorithm:

- Try them all, choose the one that is best. Not very feasible!

Feasible algorithms:

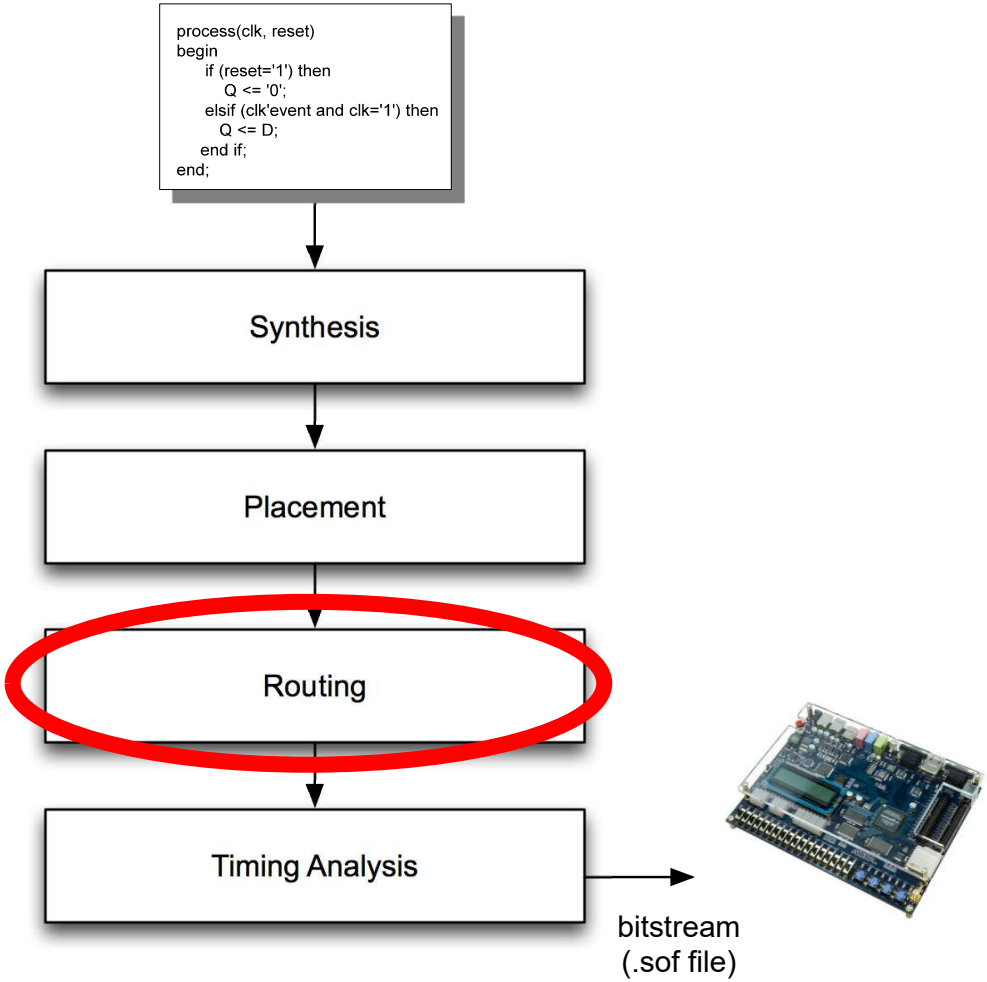
- A *Heuristic* that uses an iterative approach to find a good solution, but not necessarily the best
- Even a good heuristic can take hours to find a good solution for a large circuit

Technical details (don't worry about this for exam, but if you want to do more reading, start here):

- Altera uses algorithm called Simulated Annealing (wikipedia it)
- Xilinx uses algorithm based on quadratic optimization

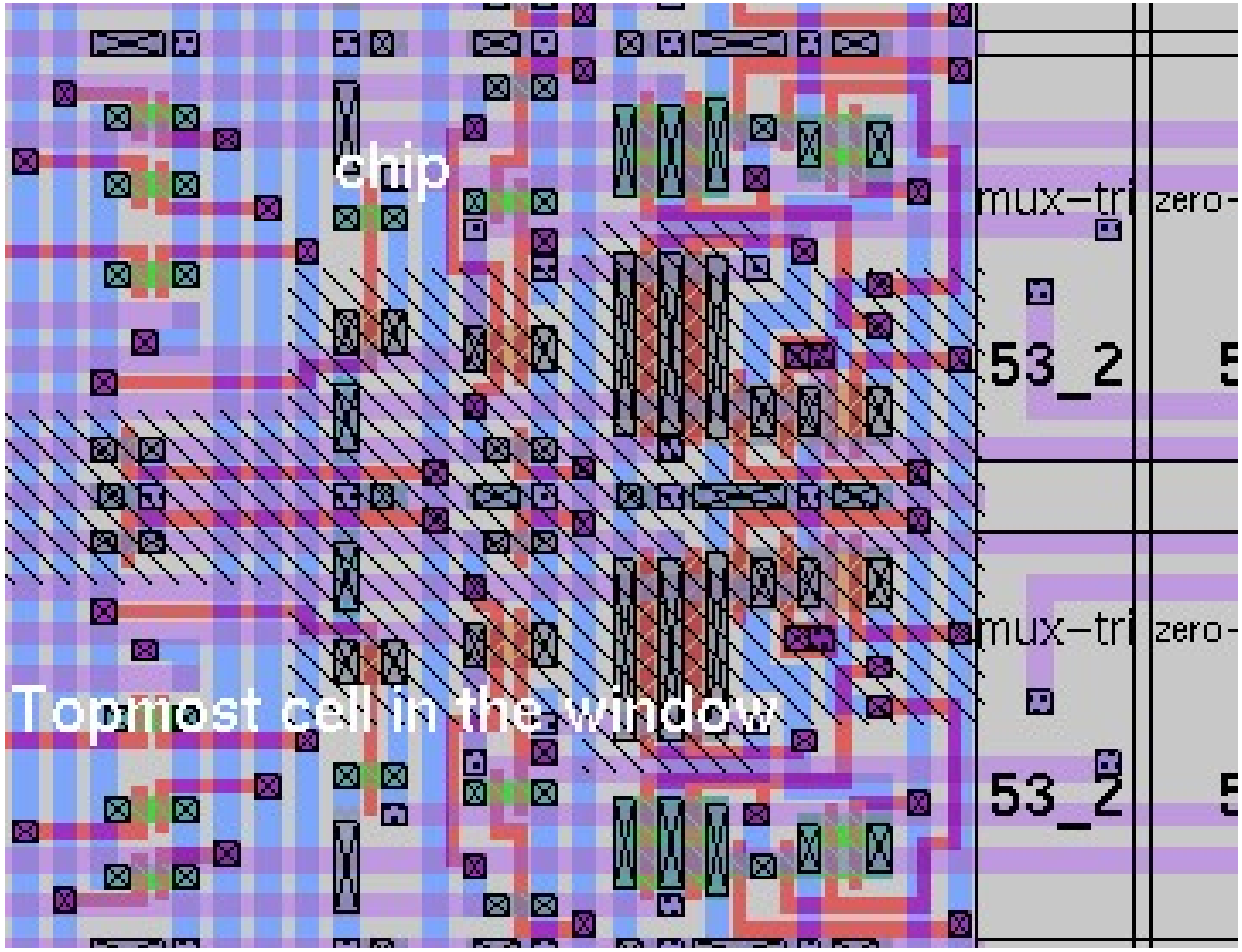
FPGA Compilation:

Next step is routing,
in which the tool finds a
path for each connection
between logic blocks



Connecting Logic Blocks

In a custom chip, you can create metal lines on the substrate to connect your gates.



This only works because the connection pattern is known at the time the chip is fabricated.

In an FPGA, the chip is used to implement many different user circuits, and the connection pattern would be different for each of these circuits.

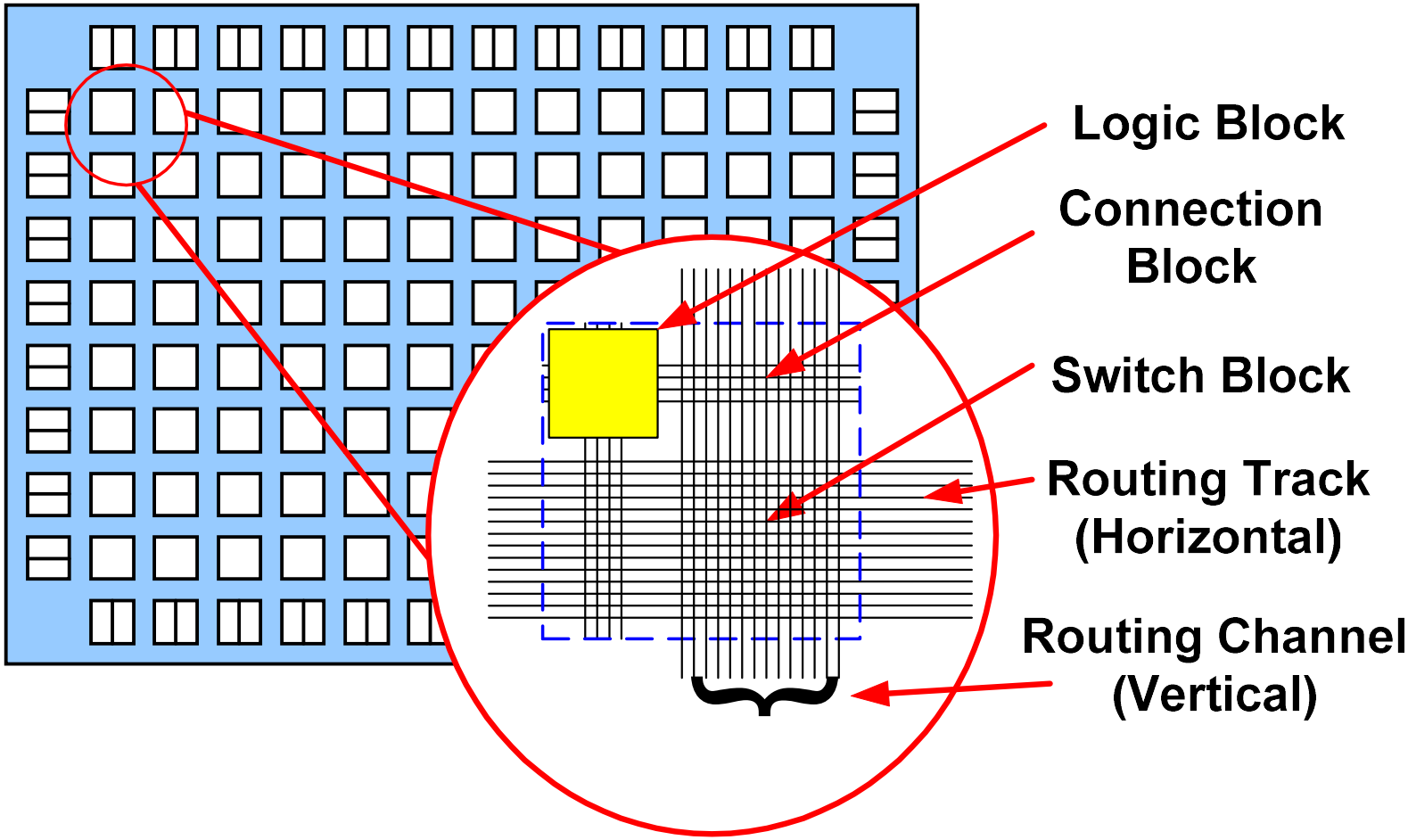
Need a way to “programmably” connect logic blocks on the chip

- Need to be able to quickly change them when you download a new user circuit

Key idea: *fixed metal tracks* (that do not change) connected using *programmable switches* (that can be configured depending on the user circuit)

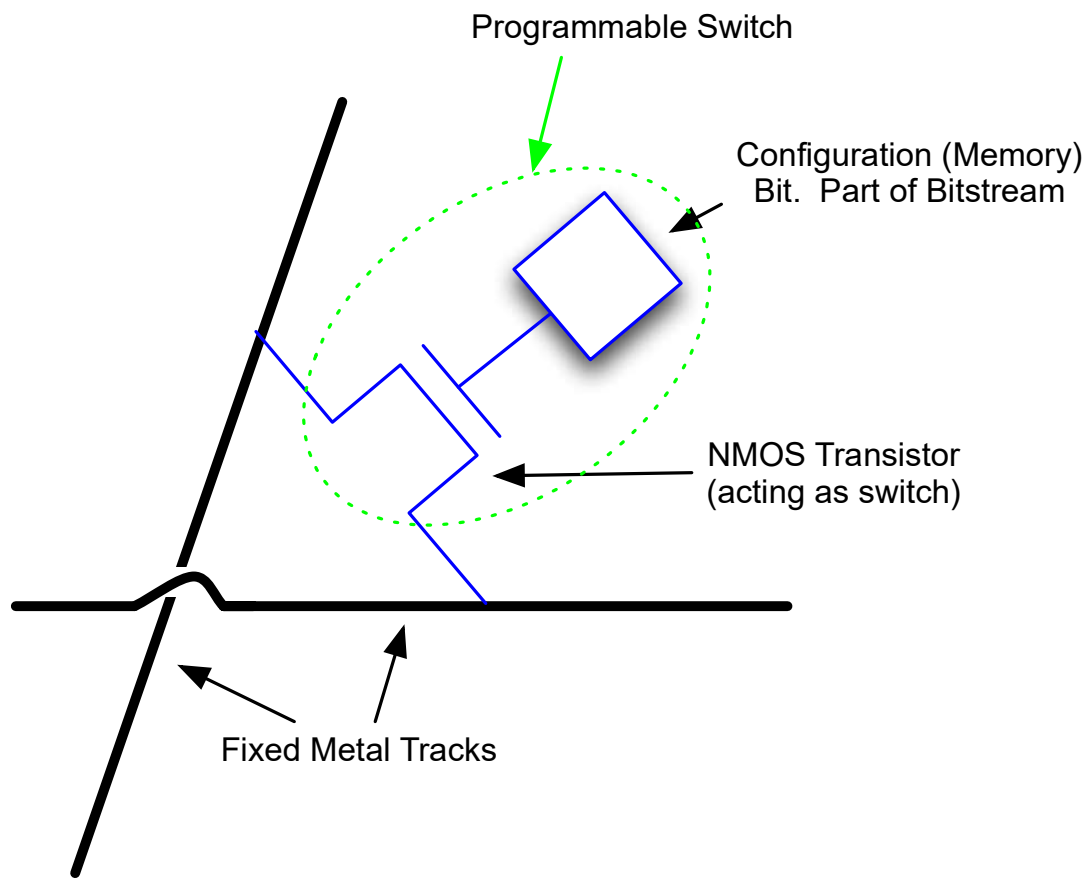
Routing Fabric

What's Inside an FPGA?



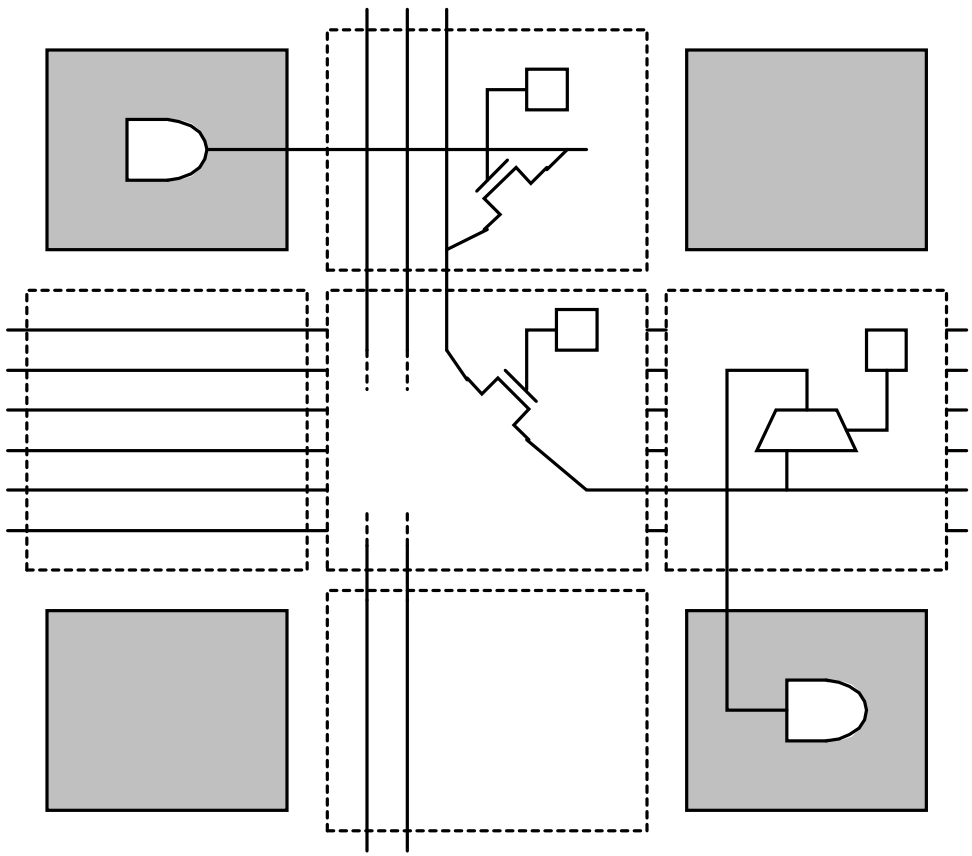
Programmable Switch

NMOS acts as a conductor when configuration bit is 1 and an open circuit when configuration bit is 0



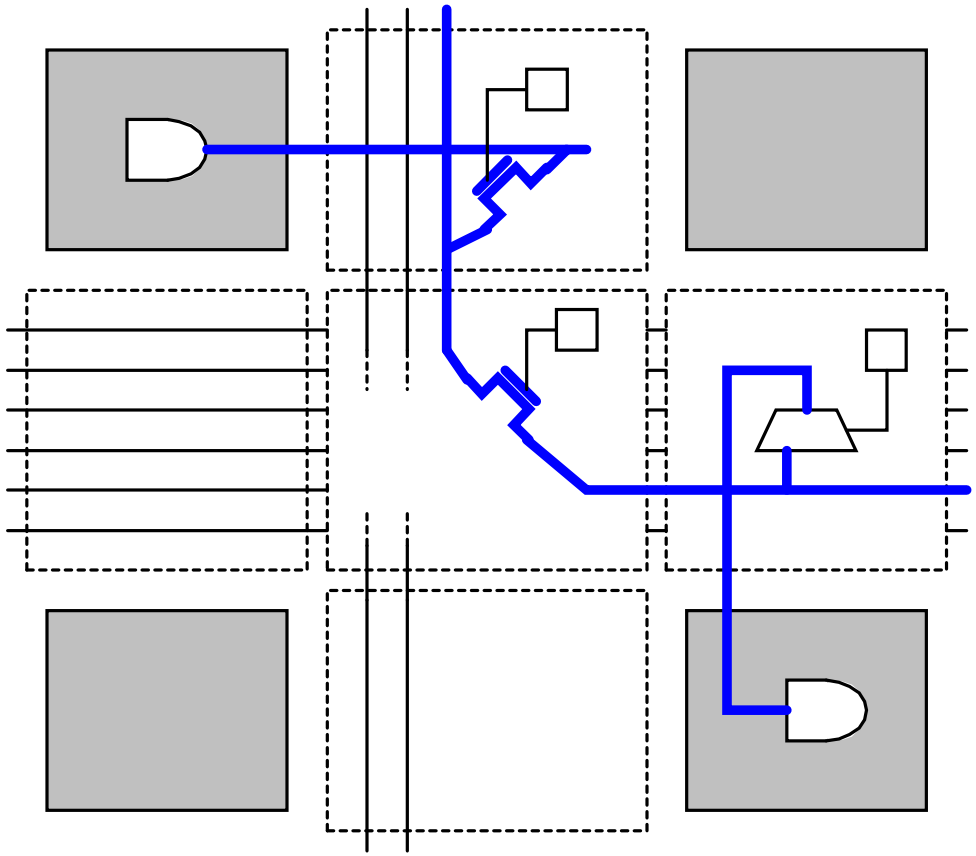
Depending on configuration bit, these two fixed metal tracks may be connected or not connected

Reconfigurable Logic:



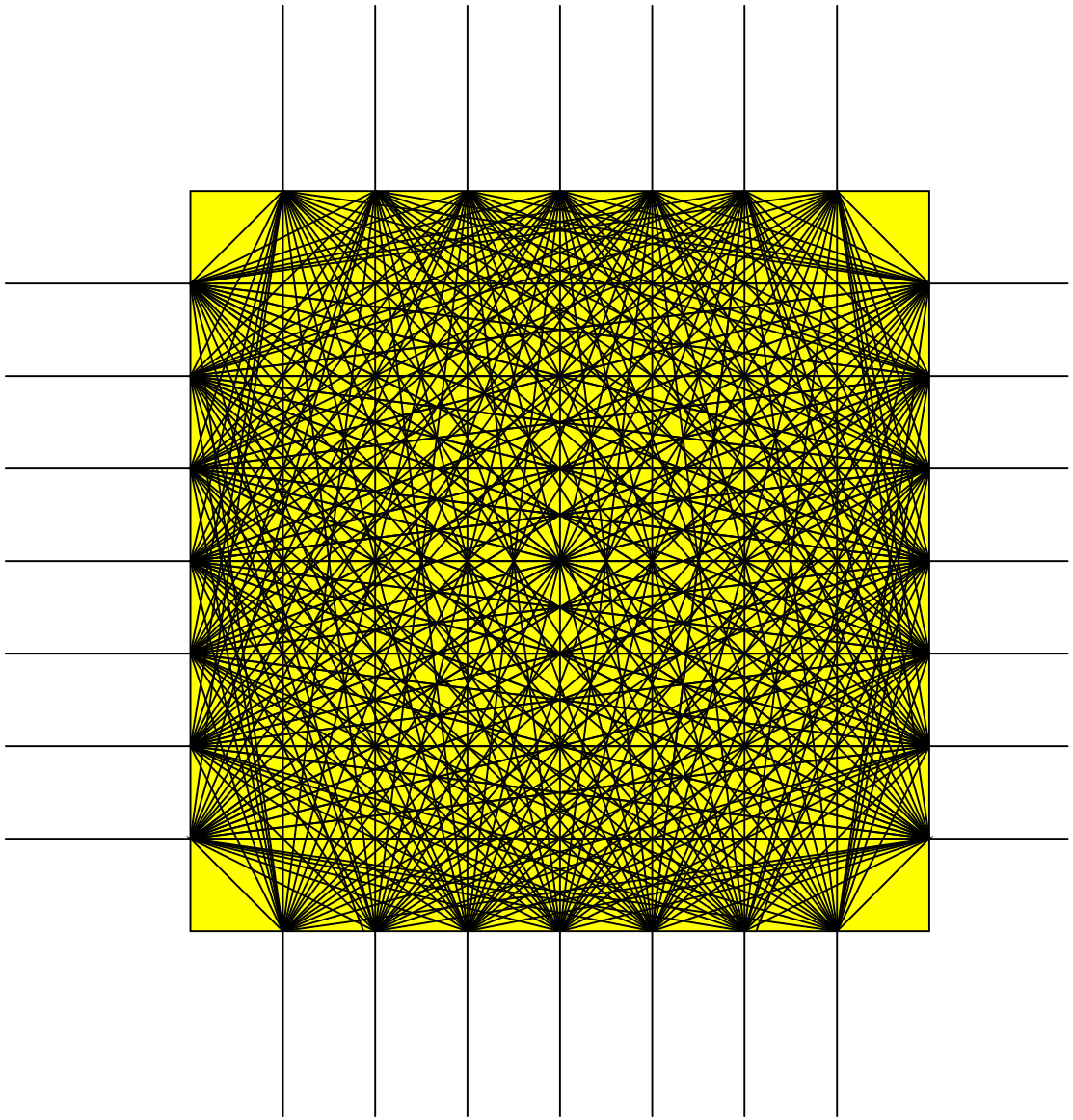
Connect Logic Blocks using Fixed Metal Tracks and Programmable Switches

Reconfigurable Logic:



Connect Logic Blocks using Fixed Metal Tracks and Programmable Switches

Switch Blocks:

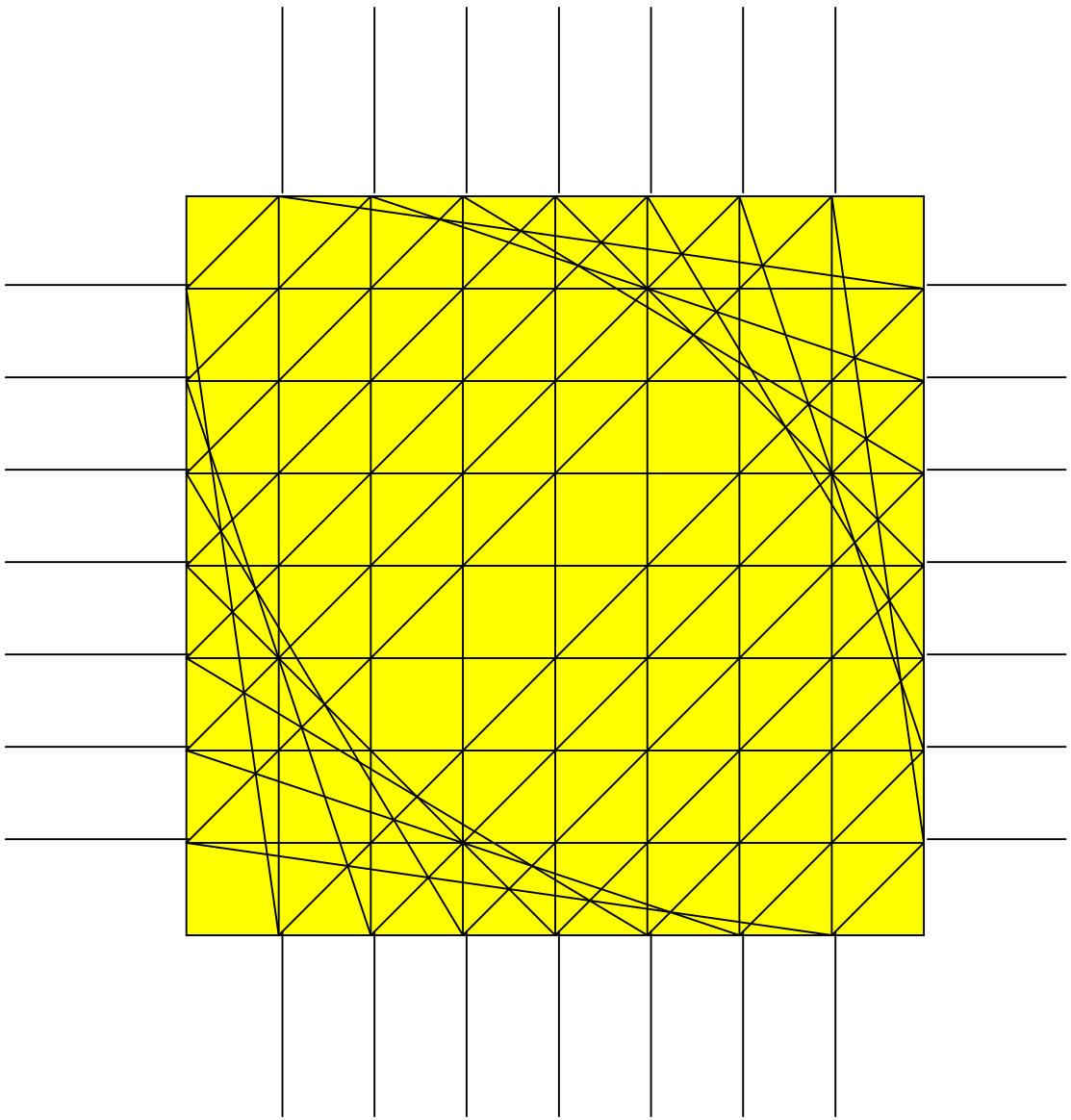


Switch Blocks connect horizontal and vertical channels

Every possible connection?

- Too big
- Too slow

Switch Blocks:



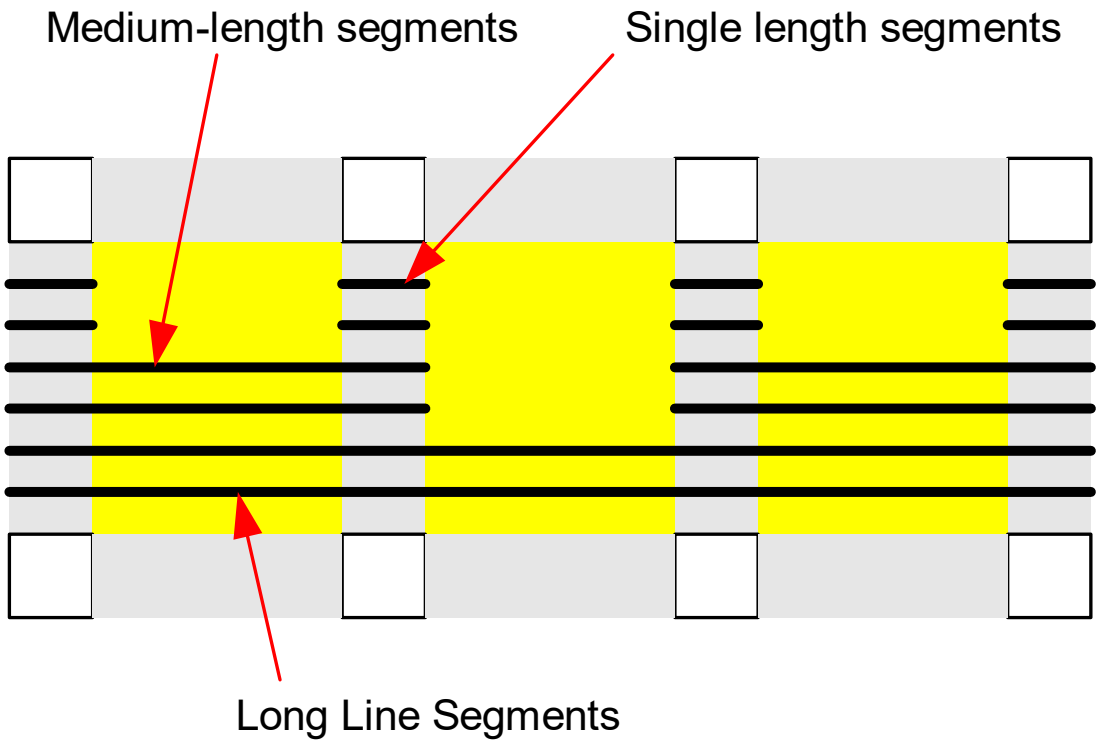
Switch Blocks connect horizontal and vertical channels

Every possible connection?

- Too big
- Too slow

Many Topologies possible

Wiring Segments



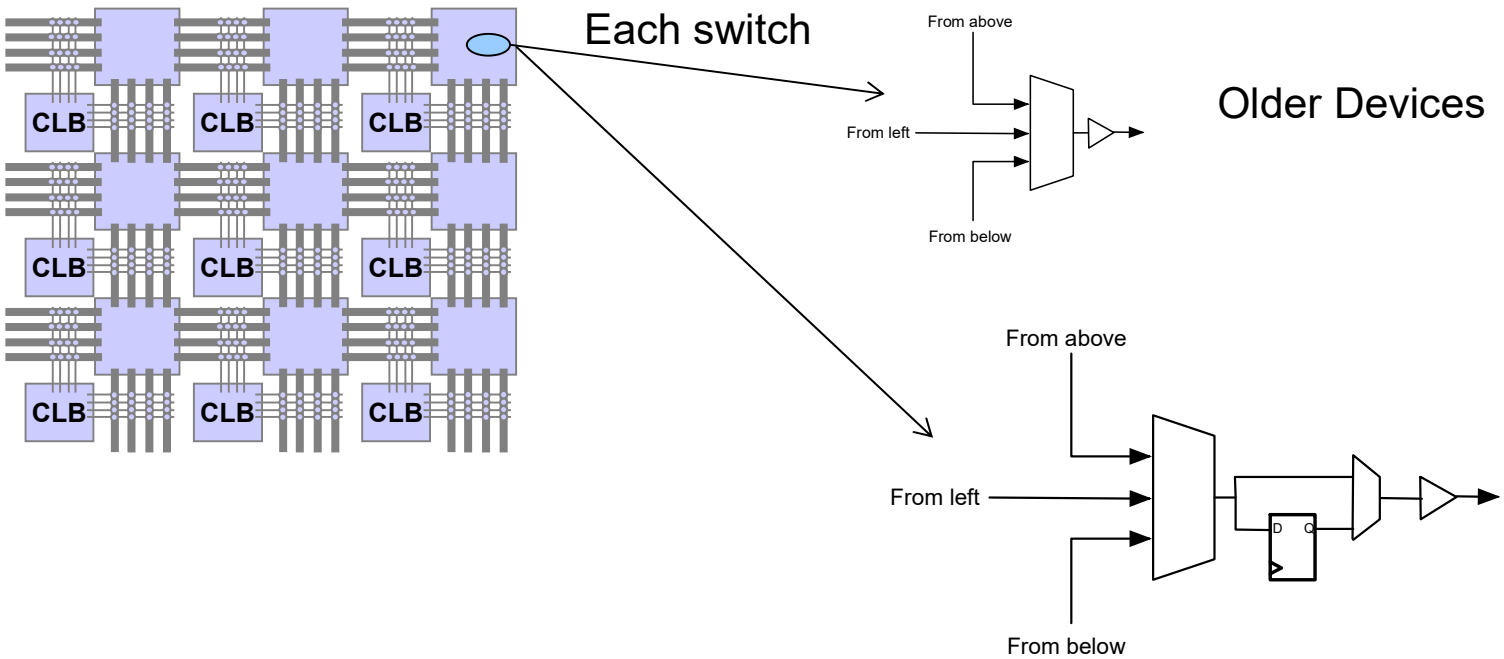
Short segments are good for local connections

Long segments are good for global connections

Most FPGA's have a variety of segment lengths

Hyper-Pipelining Stratix 10

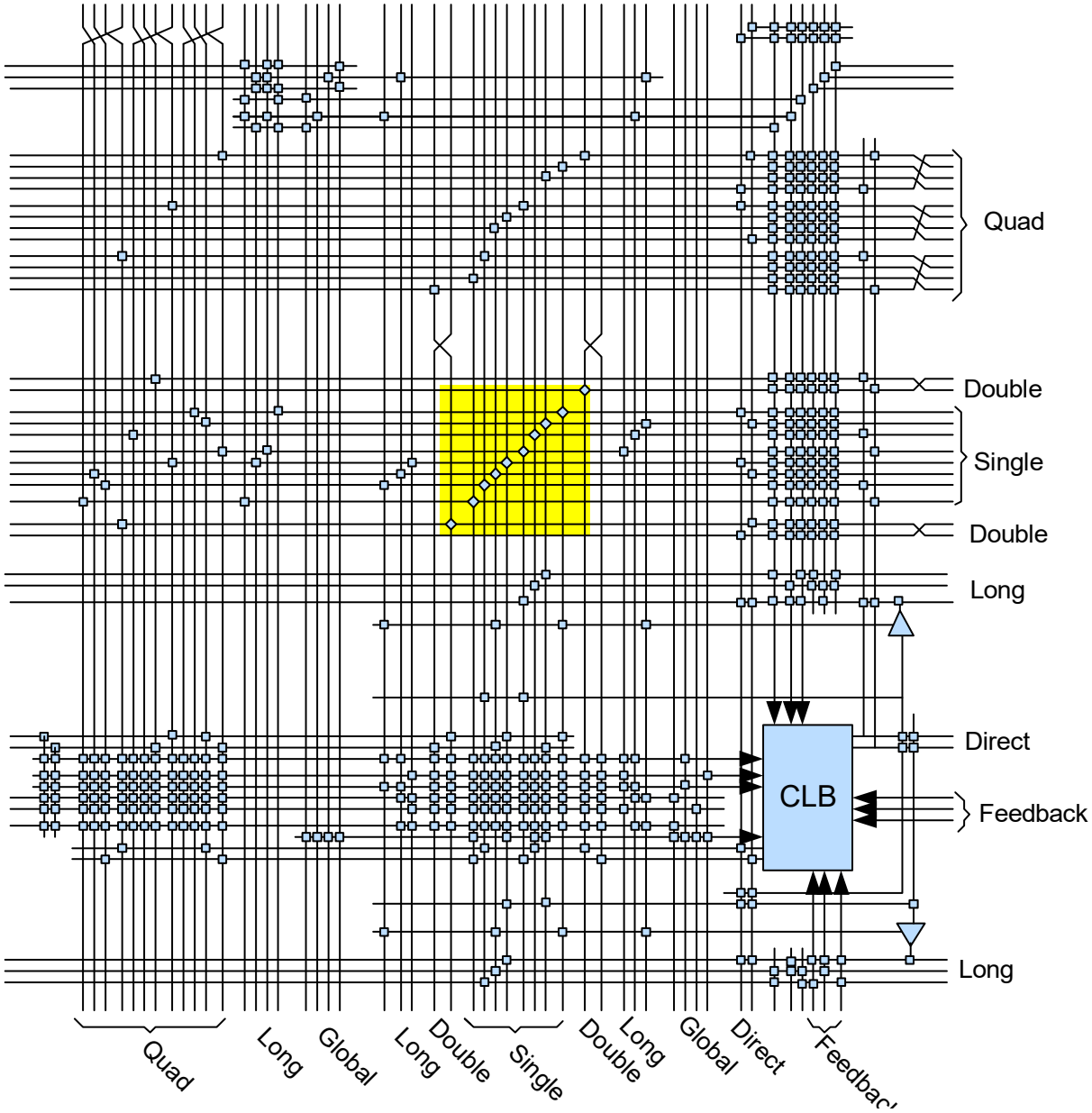
Most recent device has an optional flip-flop at *every switch* in the routing.



Devices supporting
Hyper-Pipelining (Stratix 10)

Important Point: Modern FPGAs support very fast clock speeds and heavily pipelined architectures. Lots of registers.

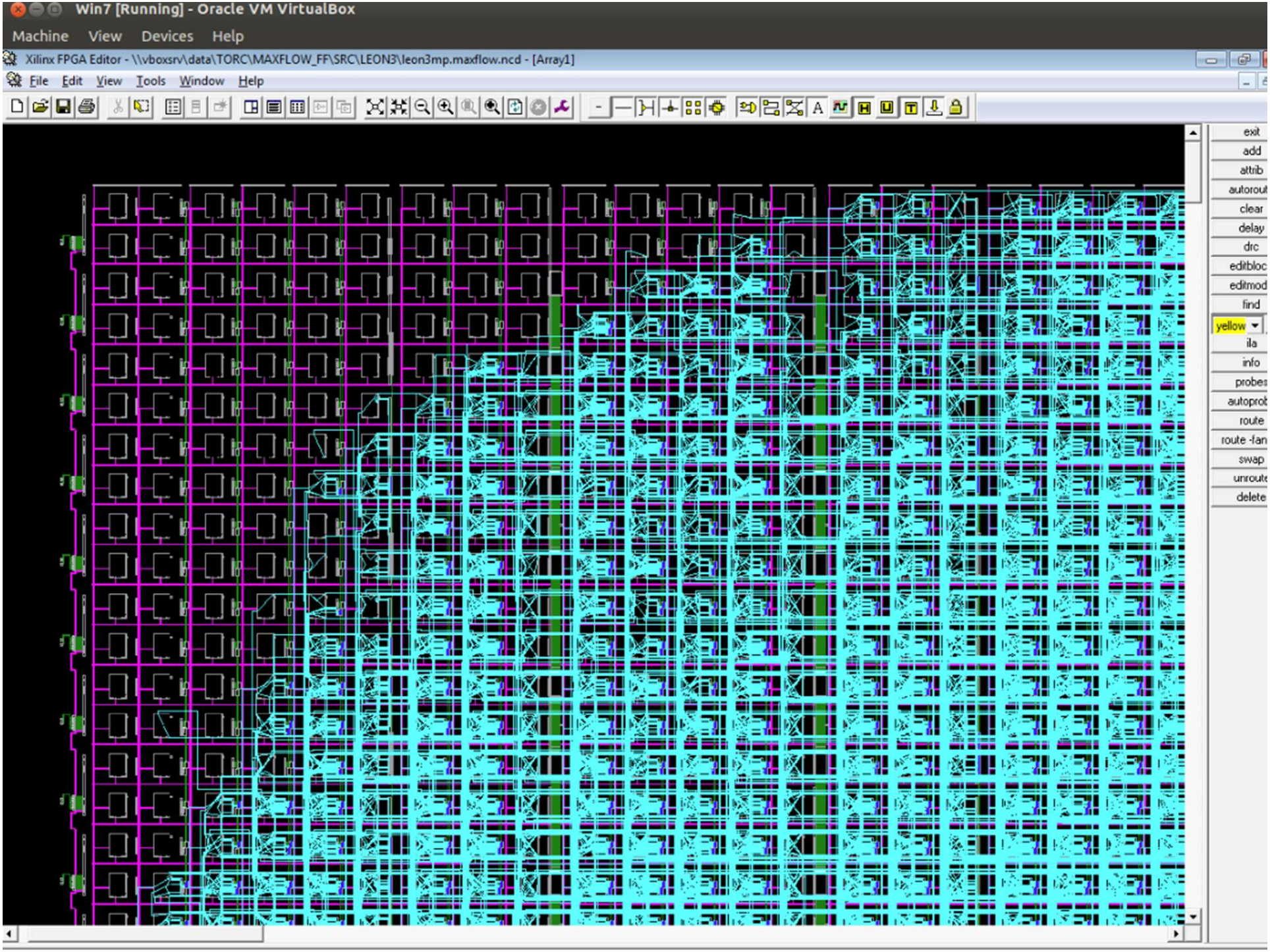
Detailed Routing Diagram (XC4000X)

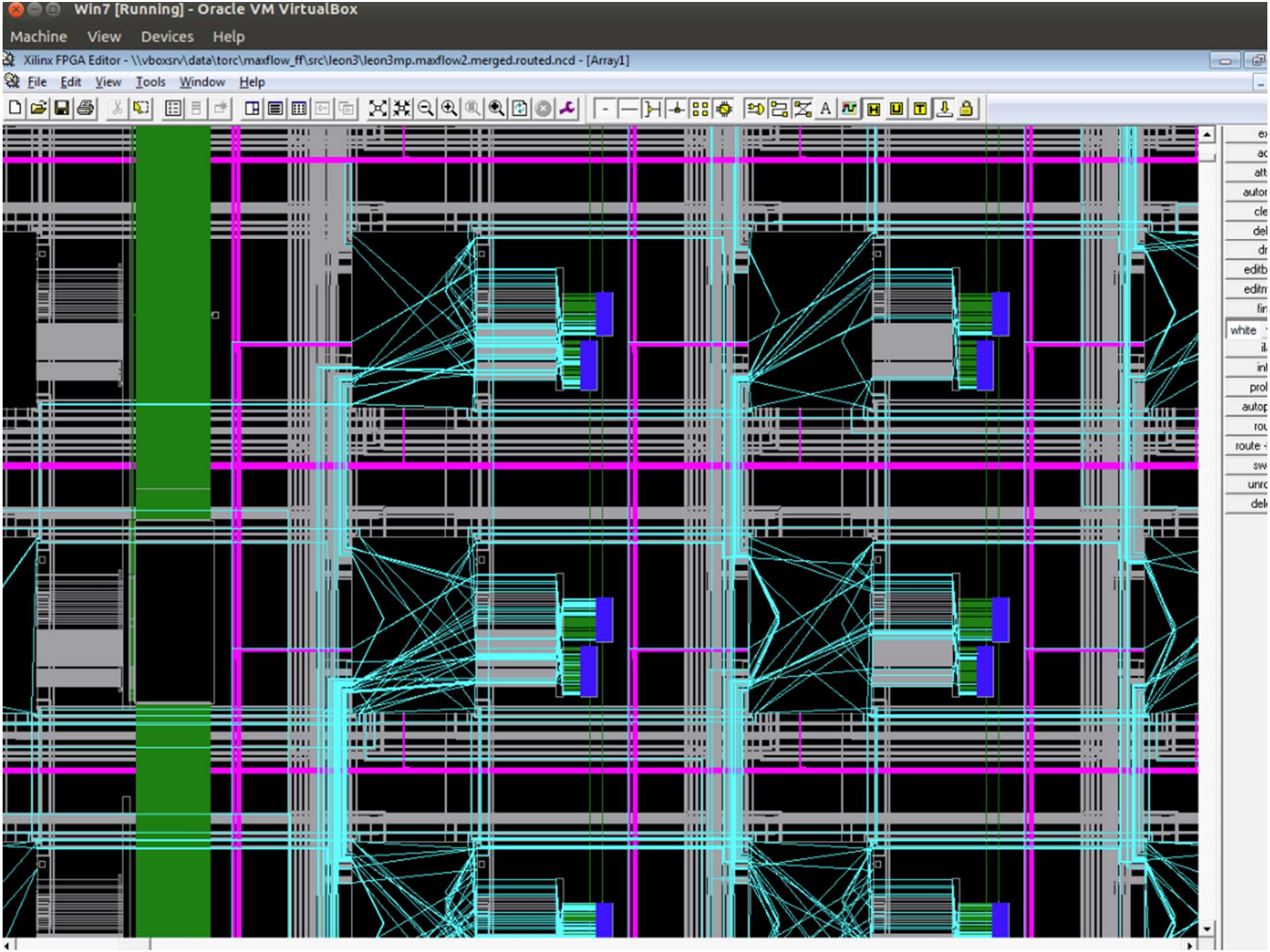


Dots represent Programmable Connections

Yes, this is old, but it illustrates the parts.

Today, vendors don't publish the routing details


Win7 [Running] - Oracle VM VirtualBox
Machine View Devices Help
Xilinx FPGA Editor - \\vboxsrv\data\TORC\MAXFLOW_FF\SRC\LEON3\leon3mp.maxflow.ncd - [Array1]
File Edit View Tools Window Help
exit
add
attrib
autorout
clear
delay
drc
editbloc
editmod
find
yellow
ila
info
probes
autoprob
route
route -fan
swap
unroute
delete


Win7 [Running] - Oracle VM VirtualBox
Machine View Devices Help
Xilinx FPGA Editor - \\vboxsrv\data\torc\maxflow_ff\src\leon3\leon3mp.maxflow2.merged.routed.ncd - [Array1]
File Edit View Tools Window Help

All 16 <<Search>>

Type	ID	Message
	170190	Fitter placement preparation operations ending: elapsed time is 00:00:00
	170191	Fitter placement operations beginning
	170137	Fitter placement was successful
	170192	Fitter placement operations ending: elapsed time is 00:00:01
	170193	Fitter routing operations beginning
	170195	Router estimated average interconnect usage is 0% of the available device resources
	170194	Fitter routing operations ending: elapsed time is 00:00:01
	170199	The Fitter performed an Auto Fit compilation. Optimizations were skipped to reduce compilat
	11888	Total time spent on timing analysis during the Fitter is 0.66 seconds.
	306004	Started post-fitting delay annotation

Routing

Routing:

During routing the tool finds legal paths for each wire in the design.

- A path is a series of fixed metal tracks connected by prog. Switches

No metal track can be used to implement two connections at once

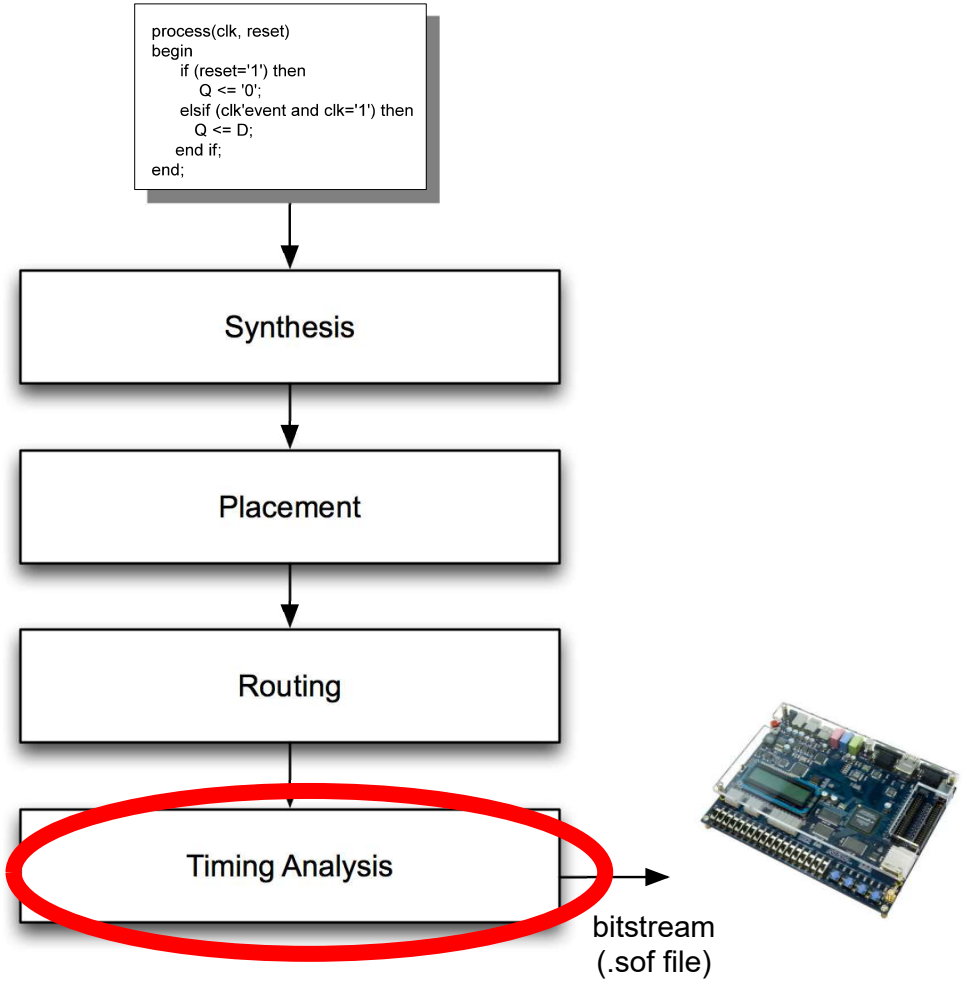
- Sometimes, routing one connection can block other connections
- Need some sort of iterative optimization procedure
- Want to give priority to connections on the critical path

Like placement, this can be time consuming:

- Can take several hours on a big design

FPGA Compilation:

During timing analysis, the circuit is analyzed to determine the maximum clock speed



All 16 <<Search>>

Type	ID	Message
	115030	Assembler is generating device programming files
		Quartus II 32-bit Assembler was successful. 0 errors, 0 warnings
	293026	Skipped module PowerPlay Power Analyzer due to the assignment FLOW_ENAB
		**********
		Running Quartus II 32-bit TimeQuest Timing Analyzer
		Command: quartus_sta sound -c sound
		qsta_default_script.tcl version: #1
	20028	Parallel compilation is not licensed and has been disabled
	21077	Low junction temperature is 0 degrees C
	21077	High junction temperature is 85 degrees C
	332104	Reading SDC File: 'sound.sdc'

Timing analysis starts here

Timing Analysis

During timing analysis.

- Finds the delay of all paths
- Determines the critical path, looks for timing violations

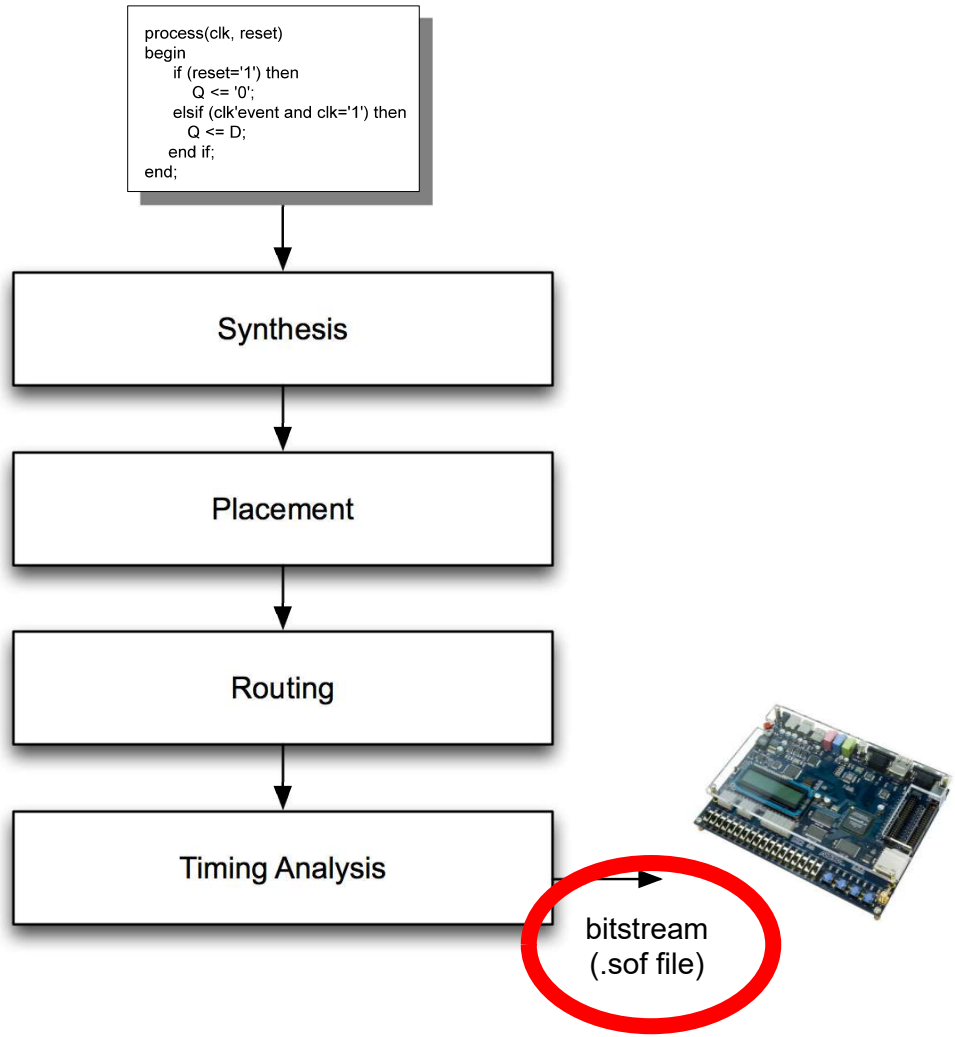
Delay of a path is the sum of:

- Delays of all logic along the path
- Delays of wiring: depends on number of programmable switches

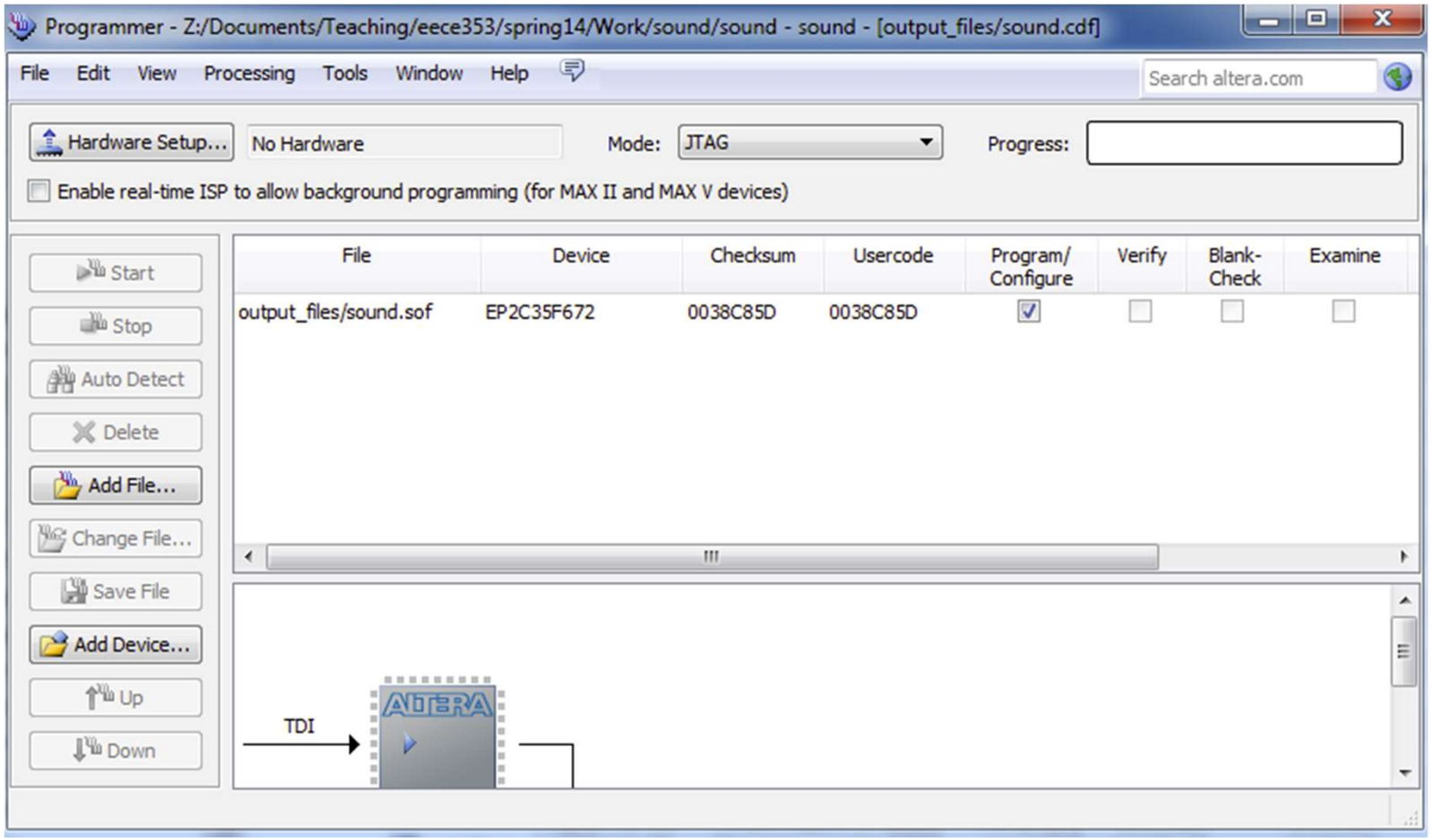
FPGA Compilation:

Last step: construct bitstream (.sof) file:

-Contains settings for conf. bits in all logic blocks
-Contains settings for conf. bits in all programmable switches



And you are ready to download your circuit…



Programming Technology

Configuration bits can be built several ways:

Memory bits (Xilinx, Intel…):

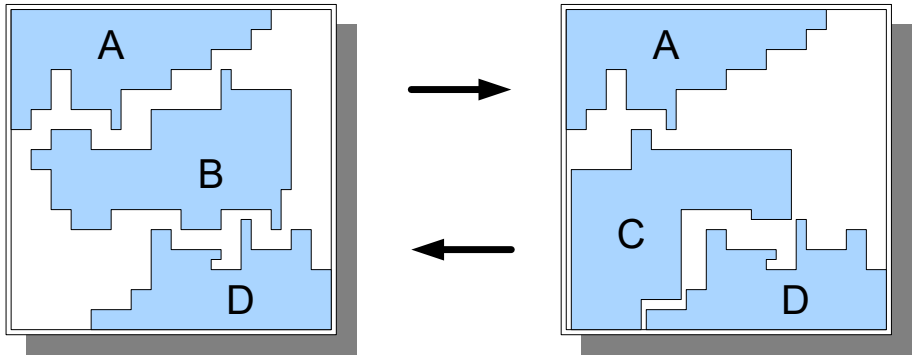
- Volatile: need to reconfigure when you power up
- Flexible: can change during run-time

Flash bits (Microchip):

- Non-Volatile: no need to reconfigure when you power up
- Limited write-cycle lifetime
- Smaller, but often a process technology behind

Advanced: Dynamic Reconfiguration

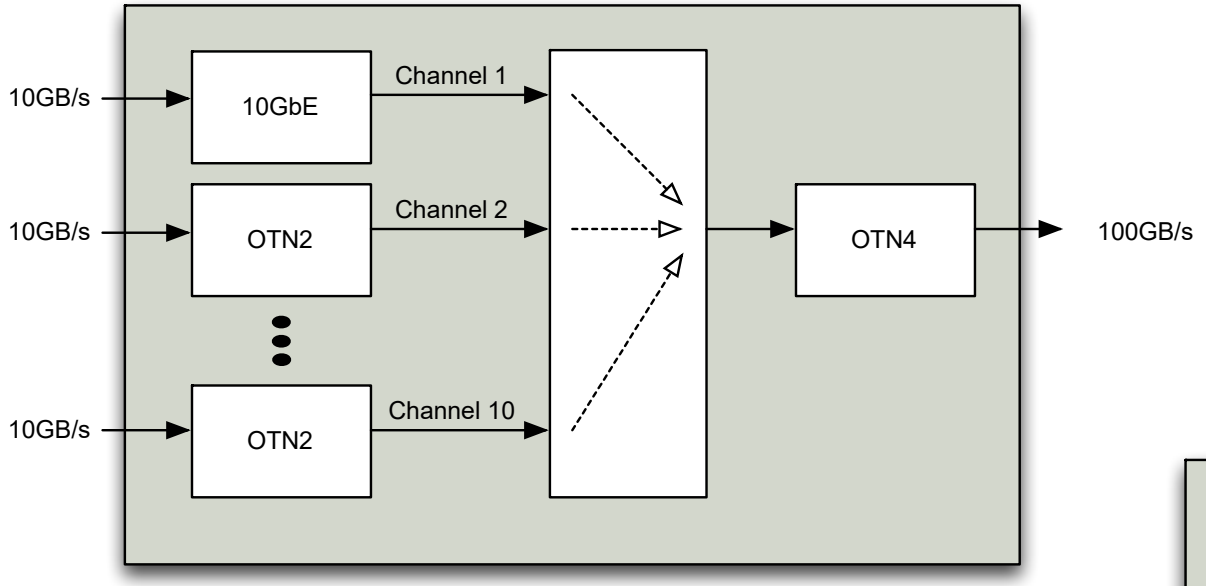
Can reprogram parts of the FPGA while other parts are still running!

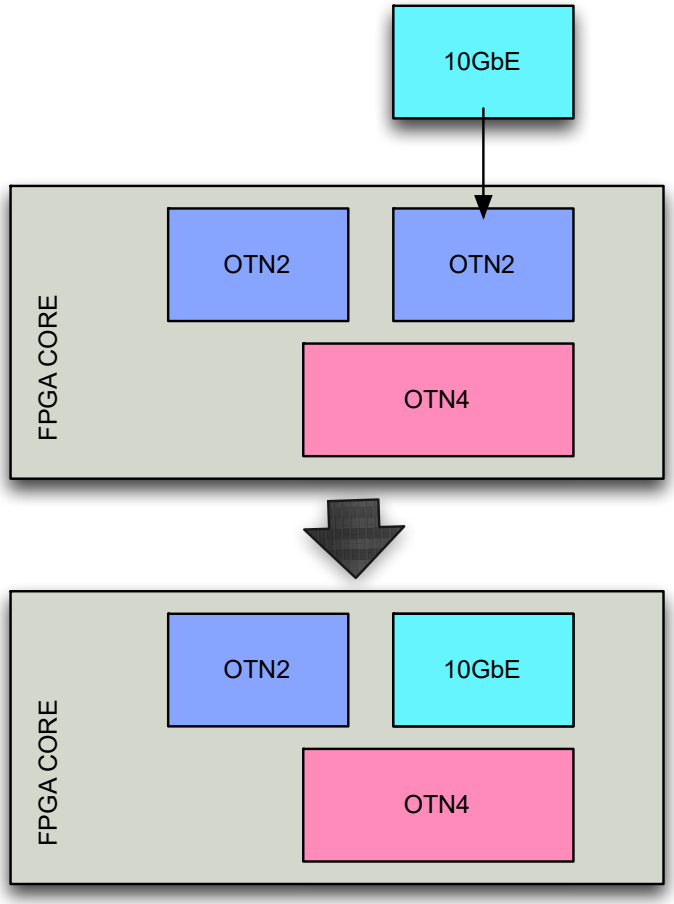


In this case, we can replace B with C without touching A or D

CAD tool support immature, not done in industry too often

Example Application of Partial Reconfiguration:





Dynamic Partial Configuration

Although DPR has been successfully used many times…

The “Killer App” for DPR has yet to emerge

In most real industrial projects with tight timelines, the extra design time and cost is not worth it

- Almost always better to just get a bigger chip

FPGAs in Datacenters: Is this the killer app???

Embedded Processors:

Embedded Processors can be “soft” or “hard”:

Soft: Processor implemented using the LUTs + memory of the FPGA

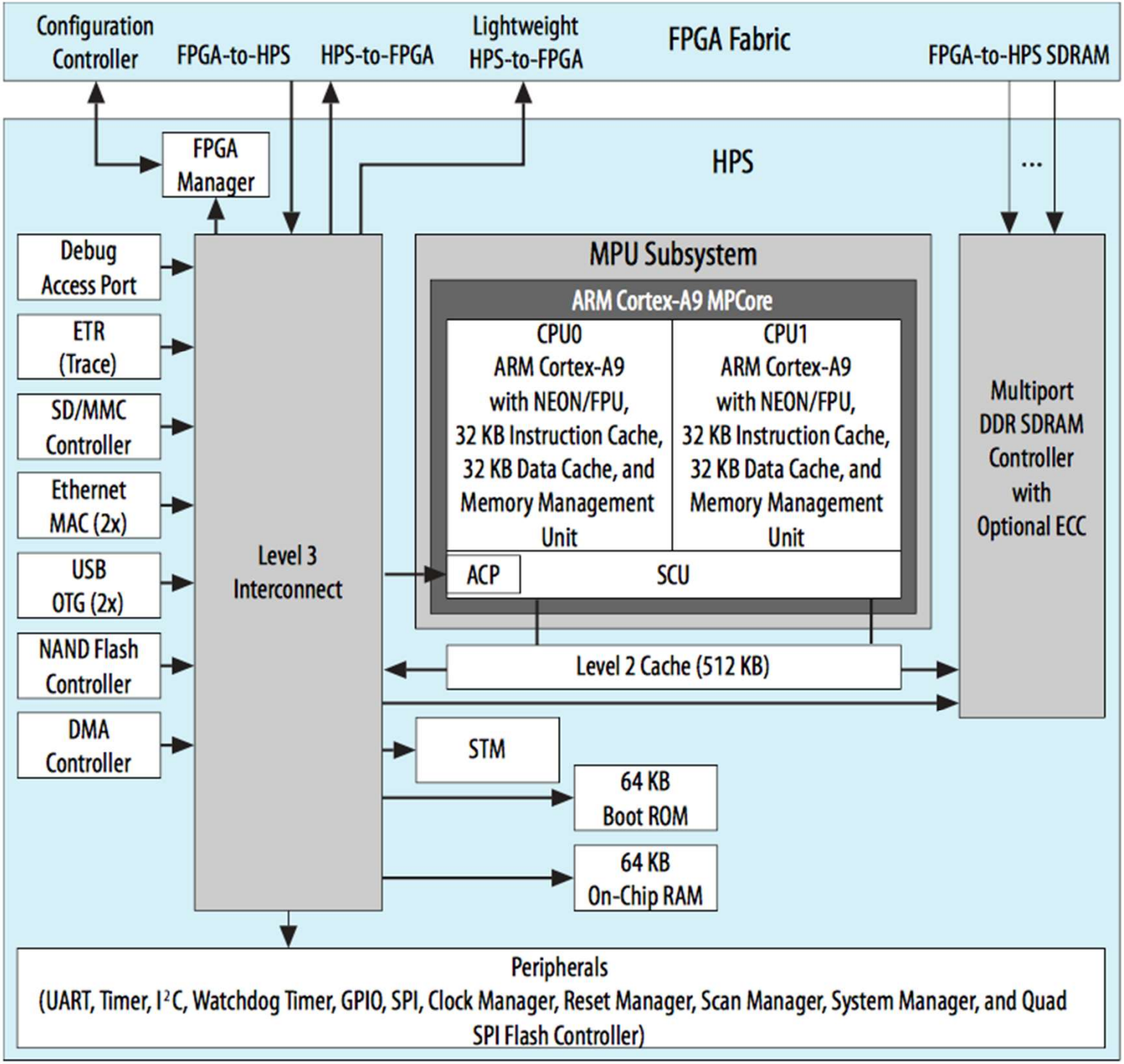
Intel: NIOS-II
Xilinx: MicroBlaze

Hard: Hard macro embedded into an FPGA (eg. Cyclone V on DE1-Soc)

Intel: ARM Cortex-A9, A53, maybe x86 some day?
Xilinx: ARM Cortex-A9, -A53, -A72, -R5, (previously: PowerPC 440)

Processors can also be coupled to an FPGA

Cyclone V:



From Altera, Cycle V Handbook, Jan 2015.

Summary

Two Sources of Flexibility in an FPGA:

1. Most FPGAs use Lookup-Tables as their basic logic resource
   - 4-LUT can implement any function of 4 inputs
2. Connections between logic blocks can be made using fixed metal tracks
   - these fixed tracks are connected to each other and to the logic blocks using programmable switches

Learning Objectives

After this slide set in lectures you should be able to:

- Understand what is inside an FPGA
- Understand the two sources of flexibility in an FPGA:
    1. Logic block behaviour can be configured
    2. Connections between logic blocks can be configured
- Show how a digital circuit can be implemented using FPGA logic blocks
- Understand what synthesis, placement, and routing are